

ORC FILE COPY

(2)

# NAVAL POSTGRADUATE SCHOOL

## Monterey, California

AD-A205 138



DTIC  
ELECTED  
MAR 10 1989  
S D  
a H

# THESIS

UTILIZATION OF A KALMAN OBSERVER WITH  
LARGE SPACE STRUCTURES

by

Bruce M. Jackson

December 1988

Thesis Advisor

Jeff B. Burl

Approved for public release; distribution is unlimited.

189 309 030

Unclassified

security classification of this page

AD H 255-135

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified		1b Restrictive Markings	
2a Security Classification Authority		3 Distribution Availability of Report Approved for public release; distribution is unlimited.	
2b Declassification Downgrading Schedule			
4 Performing Organization Report Number(s)		5 Monitoring Organization Report Number(s)	
6a Name of Performing Organization Naval Postgraduate School	6b Office Symbol (if applicable) 33	7a Name of Monitoring Organization Naval Postgraduate School	
6c Address (city, state, and ZIP code) Monterey, CA 93943-5000		7b Address (city, state, and ZIP code) Monterey, CA 93943-5000	
8a Name of Funding Sponsoring Organization	8b Office Symbol (if applicable)	9 Procurement Instrument Identification Number	
8c Address (city, state, and ZIP code)		10 Source of Funding Numbers Program Element No   Project No   Task No   Work Unit Accession No	
11 Title (include security classification) UTILIZATION OF A KALMAN OBSERVER WITH LARGE SPACE STRUCTURES			
12 Personal Author(s) Bruce M. Jackson			
13a Type of Report Master's Thesis	13b Time Covered From To	14 Date of Report (year, month, day) December 1988	15 Page Count 85
16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
17 Cosati Codes		18 Subject Terms (continue on reverse if necessary and identify by block number) Kalman Observer; large space structures. (18)	
Field	Group	Subgroup	
19 Abstract (continue on reverse if necessary and identify by block number) Control of the motions and vibrations of large space structures require the knowledge of state values that may not be available due either to inability to measure the states or, the high cost of the sensors to measure the required states. One solution is the use of an observer to estimate the states from limited sensor input. The physical characteristics of large space structures and the environment they operate in will cause large amounts of noise in the measurements. The obvious observer for such an environment is the Kalman Filter which is specifically designed to produce optimal estimates in a noisy environment. A straightforward application of the Kalman Filter will be examined utilizing a steady state Kalman gain matrix. The observer performance will be examined in both matched filter/plant and reduced order filter configurations.			
20 Distribution Availability of Abstract <input checked="" type="checkbox"/> unclassified unlimited <input type="checkbox"/> same as report <input type="checkbox"/> DTIC users		21 Abstract Security Classification Unclassified	
22a Name of Responsible Individual Jeff B. Burl		22b Telephone (include Area code) (408) 646-2390	22c Office Symbol 62BL

• DD FORM 1473,84 MAR

83 APR edition may be used until exhausted  
All other editions are obsolete

security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

Utilization of a Kalman Observer with Large Space Structures

by

Bruce M. Jackson  
Lieutenant Commander, United States Navy  
B.S., United States Naval Academy, 1976

Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL  
December 1988

Author:

Bruce M. Jackson  
Bruce M. Jackson

Approved by:

Jeff B. Burl  
Jeff B. Burl, Thesis Advisor

H. A. Titus  
H. A. Titus, Second Reader

John P. Powers  
John P. Powers, Chairman,  
Department of Electrical and Computer Engineering

Gordon E. Schacher  
Gordon E. Schacher,  
Dean of Science and Engineering

## ABSTRACT

Control of the motions and vibrations of large space structures require the knowledge of state values that may not be available due either to inability to measure the states or, the high cost of the sensors to measure the required states. One solution is the use of an observer to estimate the states from limited sensor input.

The physical characteristics of large space structures and the environment they operate in will cause large amounts of noise in the measurements. The obvious observer for such an environment is the Kalman Filter which is specifically designed to produce optimal estimates in a noisy environment.

A straightforward application of the Kalman Filter will be examined utilizing a steady state Kalman gain matrix. The observer performance will be examined in both matched filter/plant and reduced order filter configurations.

<b>Accession For</b>	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
<b>Justification</b>	
<p>For ...</p> <p>DTIC Distribution</p> <p>Approved by (Signature) 103</p> <p>Approved by (Signature)</p> <p>Approved by (Signature)</p>	



## THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

## TABLE OF CONTENTS

I. INTRODUCTION .....	1
A. BACKGROUND .....	1
B. PROBLEM STATEMENT .....	1
C. ORGANIZATION .....	2
II. MATHEMATICAL MODEL .....	3
A. INTRODUCTION .....	3
B. MODAL MODEL .....	3
C. DISCRETE-TIME MODEL .....	6
III. THE OBSERVER .....	10
A. INTRODUCTION .....	10
B. KALMAN FILTER EQUATIONS .....	10
C. STEADY-STATE SOLUTION .....	12
D. OBSERVER PERFORMANCE .....	12
IV. SIMULATION .....	14
A. INTRODUCTION .....	14
B. PLANT AND OBSERVER DATA .....	14
C. SIMULATION PROGRAMS .....	14
D. EFFECT OF PLANT TO MEASUREMENT NOISE RATIO ON OBSERVER PERFORMANCE .....	15
E. EFFECTS OF INCREASED MODES ON OBSERVER PERFORMANCE .....	16
F. REDUCED ORDER KALMAN OBSERVER .....	16
V. CONCLUSIONS AND RECOMMENDATIONS .....	35
A. CONCLUSIONS .....	35
B. RECOMENDATIONS .....	35
APPENDIX A. KALMAN GAIN MATRIX GENERATION PROGRAM .....	36

APPENDIX B. KALMAN OBSERVER AND PLANT SIMULATION .....	49
APPENDIX C. PROGRAM TO ESTIMATE NOISE IN KALMAN FILTER FROM UNOBSERVED MODES .....	62
LIST OF REFERENCES .....	73
INITIAL DISTRIBUTION LIST .....	74

## LIST OF TABLES

Table 1. CUMULATIVE UNANTICIPATED NOISE FROM UNOBSERVED MODES .....	18
Table 2. UNANTICIPATED NOISE FROM UNOBSERVED MODES BY MODE .....	18

## LIST OF FIGURES

Figure 1. Observer Performance (J) PN/MN = 1.0d12 .....	19
Figure 2. Observer Performance (J) PN/MN = 1.0d11 .....	20
Figure 3. Observer Performance (J) PN/MN = 1.0d10 .....	21
Figure 4. Observer Performance (J) PN/MN = 5.0d09 .....	22
Figure 5. Observer Performance (J) PN/MN = 2.5d09 .....	23
Figure 6. Observer Performance (J) PN/MN = 1.0d09 .....	24
Figure 7. Mode 7 (Postion) Observer Performance versus PN/MN .....	25
Figure 8. Settling Time versus PN/MN .....	26
Figure 9. Observer Performance (J) 4 Modes (7 - 10) .....	27
Figure 10. Observer Performance (J) 5 Modes (7 - 11) .....	28
Figure 11. Observer Performance (J) 6 Modes (7 - 12) .....	29
Figure 12. Observer Performance (J) 7 Modes (7 - 13) .....	30
Figure 13. Observer Performance (J) 8 Modes (7 - 14) .....	31
Figure 14. Observer Performance (J) 9 Modes (7 - 15) .....	32
Figure 15. Observer Performance (J) 10 Modes (7 - 16) .....	33
Figure 16. Settling Time versus number of Modes Observed .....	34

## ACKNOWLEDGEMENTS

I would like to express my appreciation to the McDonnel Douglas Astronautics Company of Huntington Beach, California, for allowing the use of their dynamic model of a preliminary space station configuration. Also, I would like to thank Major William J. Preston, USMC for allowing the use of the mathematical model for the station dynamics which makes up the majority of Chapter II of this paper as well as portions of his simulation program. Finally, I would like to thank Professor J. B. Burl for his assistance and most of all his patience in the completion of this work.

This paper is dedicated to Christina Marie Jackson (USNA '10).

## I. INTRODUCTION

### A. BACKGROUND

The advent of large space structures poses a number of problems for the control engineer. Previously, the objects put into space could be treated as rigid bodies so that a single three axis sensor package could be used to tell the motion of all components. The large space structures will not be rigid, instead they will have considerable flexibility and multiple modes of vibration [Ref. 1: p. 51]

Control of the structure's attitude and vibrations requires knowing the motions of the components. One approach would be to heavily instrument the space structure, but weight and cost make this approach impractical. An alternative is to use a limited number of sensors to measure only certain states and to deduce the other required states by use of an observer algorithm.

This thesis will address the production of estimates of the states needed for control of the structure. The model used will be a early design study by McDonnell Douglas Astronautics for a dual keel space station. The techniques and problems of observation for this model are generic to all large space structures.

### B. PROBLEM STATEMENT

Design of an observer for estimating the states of a large space structure breaks down into several steps. First, a mathematical model is developed for the system behavior over time. Modal analysis is used to form a system composed of decoupled second order differential equations. The use of decoupled equations allows a reduced order model to be generated by truncating the number of modal equations. A reduced order model will have all of the same mathematical qualities (and problems) but reduces the amount of time and computer resources required to do simulation.

Second, the observer is designed. The observer is designed to obtain a minimum variance estimate of the desired state values from the measurements.

Third, the observer is simulated to verify performance. Simulation runs of both a matched observer/plant system and a reduced order observer are employed. That is, the system is run where the observer is used to estimate all of the plant states and run where there are more plant states than the observer estimates.

Fourth, results are analysed and conclusions drawn based on these results. Reccomendations for further areas of research are suggested based on the results and conclusions.

### **C. ORGANIZATION**

The model of the space station is developed in Chapter II. The modal model was developed using modal analysis and discretized to form the discrete-time state equations. The data for this model was from an early design study by McDonnell Douglas Astronautics Company for a dual-keel space station. The observer and its equations are developed in Chapter III. Chapter IV is the simulation runs of the observer versus the plant. Chapter V presents conculsions and recommendations for further research.

## II. MATHEMATICAL MODEL

### A. INTRODUCTION

Prior to the proposed space station almost all of the objects put into space could be treated as simple rigid bodies for the purpose of mathematical modelling of their motions. The design constraints imposed by the high cost of lifting mass to orbit dictates a light, open structure with considerable flexing. Large space structures such as the space station, therefore, cannot be treated as rigid bodies. The structure is in fact lightly damped with multiple natural frequencies. The result is a structure that will vibrate for considerable periods of time whenever external forces are applied.

The space station structure can be modeled as an n-DOF (degree of freedom) system consisting of n masses, springs, and dashpots [Ref. 2: p. 173-176]. This straight forward modelling of the coupled masses produces a system of unworkable complexity. As a result, the system will be modelled in terms of the structures natural modes of vibration. The resulting system, while still complex, is at least workable.

The model will be developed in two steps. The first will be to generate the continuous-time model of the natural modes. The second will yield the discrete-time model, developed from the first model, for use in the simulation.

### B. MODAL MODEL

The space station structure can be modeled as a system of discrete masses coupled by springs and dashpots. The major mechanism of damping in the structure is structural damping, the internal dissipation of energy within the members, as the structure vibrates. Structural damping can be shown to be equivalent to viscous damping and this equivalency is used in the model [Ref. 2: p. 72-73].

The energy dissipated by structural damping is:

$$W_d = \alpha X^2 \quad (1)$$

$W_d$  = energy dissipated by structural damping

$\alpha$  = constant (force/displacement)

$X$  = displacement

The energy dissipated by viscous damping is:

$$W_v = \pi c \omega X^2 \quad (2)$$

We can equate the two

$$\pi C_{eq} \dot{X}^2 = \alpha X^2 \quad (3)$$

yielding an equivalent viscous damping coefficient:

$$C_{eq} = \frac{\alpha}{\pi \omega} \quad (4)$$

The second order differential equation for a single viscously damped mass is:

$$m\ddot{x} + c\dot{x} + kx = F(t) \quad (5)$$

Substituting  $C_{eq}$  for  $c$

$$m\ddot{x} + \frac{\alpha}{\pi \omega} \dot{x} + kx = F(t) \quad (6)$$

For multiple mass systems  $C_{eq}$  becomes  $\frac{d}{\omega_f} K$  where  $\omega_f$  is the natural frequency of vibration.

The displacement of masses can be represented by the second order matrix differential equation [Ref. 3: p. 3-9],

$$M\ddot{q}(t) + \frac{d}{\omega_f} K\dot{q}(t) + Kq(t) = F(t) \quad (7)$$

$q$  = coordinate vector

$M$  = system mass matrix (diagonal)

$\frac{d}{\omega_f} K$  = equivalent damping

$d$  = damping coefficient

$\omega_f$  = frequency of oscillation of the system

$K$  = symmetric system stiffness matrix

$F(t)$  = system forcing function

The above equation represents a system of second order differential equations coupled through the stiffness matrix. Decoupling can be done by expressing  $q$  in terms of natural modes of vibration. The process is called modal analysis. The independent differential equations can then be treated individually. The modal equations are derived below.

First, the undamped, homogeneous form of Eq. (7)

$$\mathbf{M}\ddot{q}(t) + \mathbf{K}q(t) = 0 \quad (8)$$

is solved. Let

$$q(t) = Ax \sin(\omega t + \Theta) \quad (9)$$

$$\dot{q}(t) = Ax\omega \cos(\omega t + \Theta) \quad (10)$$

$$\ddot{q}(t) = -Ax\omega^2 \sin(\omega t + \Theta) \quad (11)$$

substituting Eq. (9) and Eq. (10) into Eq. (11)

$$[-\omega^2\mathbf{M} + \mathbf{K}]Ax \sin(\omega t + \Theta) = 0 \quad (12)$$

This equation has a non-trivial solution for all time if and only if:

$$[\mathbf{K} - \omega^2\mathbf{M}]x = 0 \quad (13)$$

Equation (12) has  $n$  combinations of  $x$  (natural mode shapes) and  $\omega$  (natural frequencies) as solutions. These can be grouped into matrices:

$$\mathbf{X} = [x_1 x_2 \dots x_n]^T \quad (14)$$

$$\Omega^2 = \text{diag}[\omega_{o1}^2 \omega_{o2}^2 \dots \omega_{on}^2] \quad (15)$$

which satisfy the equation:

$$\mathbf{KX} = \Omega^2 \mathbf{MX} \quad (16)$$

Several useful relations can be derived from Eq. (16). Premultiplying Eq. (16) by  $\mathbf{X}^T$ ,

$$\mathbf{X}^T \mathbf{KX} = \Omega^2 \mathbf{X}^T \mathbf{MX} \quad (17)$$

The eigenvectors can be normalized

$$\mathbf{X}^T \mathbf{MX} = \mathbf{I} \quad (18)$$

which yields

$$\mathbf{X}^T \mathbf{KX} = \Omega^2 \quad (19)$$

The equations of motion can be uncoupled through the linear transformation of the coordinate system

$$q(t) = \sum_{i=1}^n x_i \eta_i(t) = \mathbf{X} \eta(t) \quad (20)$$

$\mathbf{X}$  = modal matrix

$n$  = maximum number of degrees of freedom

$\eta(t)$  = transformed coordinate vector

Application of the transformation to the system Eq. (7) yields

$$\mathbf{X}^T \mathbf{M} \mathbf{X} \ddot{\eta}(t) + \frac{d}{\omega_f} \mathbf{X}^T \mathbf{K} \mathbf{X} \dot{\eta}(t) + \mathbf{X}^T \mathbf{K} \mathbf{X} \eta(t) = \mathbf{X}^T \mathbf{F}(t) \quad (21)$$

Using Eq.(18) and Eq. (19)

$$\mathbf{X}^T \mathbf{M} \mathbf{X} \ddot{\eta}(t) = \mathbf{I} \ddot{\eta}(t) = \ddot{\eta} \quad (22)$$

$$\frac{d}{\omega_f} \mathbf{X}^T \mathbf{K} \mathbf{X} \dot{\eta}(t) = \frac{d}{\omega_f} \Omega^2 \dot{\eta}(t) = d\Omega \dot{\eta} \quad (23)$$

$$\mathbf{X}^T \mathbf{K} \mathbf{X} \eta(t) = \Omega^2 \eta \quad (24)$$

therefore

$$\ddot{\eta} + d\Omega \dot{\eta} + \Omega^2 \eta = \mathbf{X}^T \mathbf{F} \quad (25)$$

Equation (25) is the modal model of uncoupled second order differential equations. The motion of the structure can be found from the modal amplitudes,  $\eta(t)$ , using Eq. (20).

### C. DISCRETE-TIME MODEL

The discrete-time state space model is found by solving the continuous-time equations. The  $i$ th equation of motion is

$$\ddot{\eta}_i(t) + d\omega_{oi} \dot{\eta}_i(t) + \omega_{oi}^2 \eta_i(t) = \mathbf{X}_i^T \mathbf{F}(t) \quad (26)$$

$\mathbf{X}_i^T$  = transpose of the  $i$ th mode shape vector

$\mathbf{F}(t)$  = torquing force applied at a point

The homogeneous solution ( $\mathbf{F}(t) = 0$ ) for Eq. (26) is [Ref. 4: p. 475-476]

$$\eta(t) = C_1 e^{-\gamma t} \cos(\omega_d t) + C_2 e^{-\gamma t} \sin(\omega_d t) \quad (27)$$

where

$$\gamma = \frac{d\omega_{ol}}{2} \quad (28)$$

$$\omega_d = \sqrt{\omega_{ol}^2 - \gamma^2} \quad (29)$$

The constants in Eq. (27) can be found by taking the derivative

$$\dot{\eta}(t) = (C_2 \omega_d - C_1 \gamma) e^{-\gamma t} \cos(\omega_d t) - (C_1 \omega_d - C_2 \gamma) e^{-\gamma t} \sin(\omega_d t) \quad (30)$$

and evaluating at  $t = 0$

$$\eta(0) = C_1 \quad (31)$$

$$\dot{\eta}(0) = C_2 \omega_d - C_1 \gamma \quad (32)$$

Solving for  $C_1$  and  $C_2$

$$C_1 = \eta(0) \quad (33)$$

$$C_2 = \frac{\dot{\eta}(0)}{\omega_d} + \frac{\eta(0)\gamma}{\omega_d} \quad (34)$$

In matrix form

$$\begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{\gamma}{\omega_d} & \frac{1}{\omega_d} \end{bmatrix} \begin{bmatrix} \eta(0) \\ \dot{\eta}(0) \end{bmatrix} \quad (35)$$

Rewriting Eq. (27) and Eq. (30) in matrix form

$$\begin{bmatrix} \eta(t) \\ \dot{\eta}(t) \end{bmatrix} = \begin{bmatrix} e^{-\gamma t} \cos(\omega_d t) & e^{-\gamma t} \sin(\omega_d t) \\ e^{-\gamma t} [\gamma \cos(\omega_d t) + \omega_d \sin(\omega_d t)] & e^{-\gamma t} [\omega_d \cos(\omega_d t) - \gamma \sin(\omega_d t)] \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \quad (36)$$

Substituting Eq. (35) into Eq. (36), the solution can be written in terms of the initial conditions

$$\begin{bmatrix} \eta(t) \\ \dot{\eta}(t) \end{bmatrix} = \begin{bmatrix} e^{-\gamma t} [\cos(\omega_d t) + \frac{\gamma}{\omega_d} \sin(\omega_d t)] & \frac{1}{\omega_d} e^{-\gamma t} \sin(\omega_d t) \\ -\frac{\omega_o}{\omega_d} e^{-\gamma t} \sin(\omega_d t) & e^{-\gamma t} [\cos(\omega_d t) - \frac{\gamma}{\omega_d} \sin(\omega_d t)] \end{bmatrix} \begin{bmatrix} \eta(0) \\ \dot{\eta}(0) \end{bmatrix} \quad (37)$$

Letting

$$\mathbf{X}_i(t) = \begin{bmatrix} \eta_i(t) \\ \dot{\eta}_i(t) \end{bmatrix} \quad (38)$$

and

$$\mathbf{A}_i = \begin{bmatrix} e^{-\gamma t} [\cos(\omega_d t) + \frac{\gamma}{\omega_d} \sin(\omega_d t)] & \frac{1}{\omega_d} e^{-\gamma t} \sin(\omega_d t) \\ -\frac{\omega_o}{\omega_d} e^{-\gamma t} \sin(\omega_d t) & e^{-\gamma t} [\cos(\omega_d t) - \frac{\gamma}{\omega_d} \sin(\omega_d t)] \end{bmatrix} \quad (39)$$

the solution can be written as

$$\mathbf{X}_i(t) = \mathbf{A}_i(t) \mathbf{X}_i(0) \quad (40)$$

where  $\mathbf{A}_i$  is the state transition matrix of the  $i$ th mode. The non-homogeneous solution is

$$\mathbf{X}_i(t) = \mathbf{A}_i(t) \mathbf{X}_i(0) + \mathbf{B}_i x_i^T \mathbf{F}(0) \quad (41)$$

where the discrete-time input matrix, for constant  $\mathbf{F}$ , is given by

$$\mathbf{B}_i = \int_0^T \mathbf{B}_i(\tau) \Gamma d\tau \quad (42)$$

and  $\Gamma = [0 \ 1]^T$  is the input matrix for the continuous-time system, and  $T$  is the sampling time. Solving Eq. (42) yields

$$\mathbf{B}_i = \begin{bmatrix} \frac{1}{\omega_o^2} [1 - e^{-\gamma T} \cos(\omega_d T) - \frac{\gamma}{\omega_d} e^{-\gamma T} \sin(\omega_d T)] \\ \frac{1}{\omega_d} e^{-\gamma T} \sin(\omega_d T) \end{bmatrix} \quad (43)$$

The discrete-time state equation for the  $i$ th equation of motion can be written as

$$\mathbf{X}_i(kT + 1) = \mathbf{A}_i(T) \mathbf{X}_i(kT) + \mathbf{B}_i(T) x_i^T \mathbf{F}(kT) \quad (44)$$

where  $\mathbf{A}_i$  and  $\mathbf{B}_i$  are evaluated at  $t = T$ . Here,

$\mathbf{X}_i$  = vector of the  $i$ th modal amplitude and the  $i$ th modal velocity

- $\mathbf{A}_i$  = ith state transition matrix
- $\mathbf{B}_i$  = ith input vector
- $x_i^T$  = transpose of the ith mode shape vector
- $\mathbf{F}$  = distributed force on the plant
- $T$  = sampling time
- $k$  = time index

Equation (44) can be expanded to include the disturbance input,  $w(kT)$  :

$$\mathbf{X}_i(kT + 1) = \mathbf{A}_i(T)\mathbf{X}_i(kT) + \mathbf{B}_i(T)x_i^T[\mathbf{F}(kT) + w(kT)] \quad (45)$$

Equation (45) is the discrete-time mathematical model describing the motion of the structure in terms of its natural modes of vibration.

### III. THE OBSERVER

#### A. INTRODUCTION

The observer design will be required to estimate the modal states in a noisy environment. Kalman filtering is the most widely used technique for accomplishing the production of state estimates in a noisy environment [Ref. 5: p.159]. The steady state Kalman filter was selected to minimize the computations during the actual plant observer operation. Use of a steady-state gain matrix for the observer allows the matrix to be computed separately from the operational observer, reducing the computer power required for the observer and allowing the algorithm to operate more rapidly.

#### B. KALMAN FILTER EQUATIONS

The discrete Kalman filter provides state estimates for the following dynamic system [Ref. 5: p. 159-162],

$$\mathbf{X}(k+1) = \mathbf{AX}(k) + \mathbf{BU}(k) + \mathbf{BnW}(k) \quad (46)$$

$$\mathbf{Y}(k+1) = \mathbf{CX}(k+1) + \mathbf{V}(k+1) \quad (47)$$

$\mathbf{X}$  =  $n \times 1$  state vector

$\mathbf{U}$  =  $P \times 1$  control vector

$\mathbf{W}$  =  $r \times 1$  plant noise vector

$\mathbf{Y}$  =  $m \times 1$  measurement vector

$\mathbf{V}$  =  $m \times 1$  measurement noise vector

$\mathbf{A}$  =  $n \times n$  state transition matrix

$\mathbf{B}$  =  $n \times p$  control input matrix

$\mathbf{Bn}$  =  $n \times r$  plant noise input matrix

$\mathbf{C}$  =  $m \times n$  measurement matrix

The plant noise vector  $\mathbf{W}(k)$  is gaussian white noise with

$$\mathbf{E}\{\mathbf{W}(k)\} = \mathbf{0} \quad (48)$$

$$\mathbf{E}\{\mathbf{W}(k)\mathbf{W}^T(k)\} = \mathbf{Q} \quad (49)$$

for all  $k = 0, 1, 2, \dots$ , and  $\mathbf{Q}$  is a positive semi-definite  $r \times r$  matrix.  $\mathbf{V}(k)$  is gaussian white noise with

$$\mathbf{E}\{\mathbf{V}(k)\} = \mathbf{0} \quad (50)$$

$$\mathbf{E}\{\mathbf{V}(k)\mathbf{V}^T(k)\} = \mathbf{R} \quad (51)$$

for all  $k = 0, 1, 2, \dots$ , and  $\mathbf{R}$  is a positive definite  $m \times m$  matrix. The two random processes  $\mathbf{W}(k)$  and  $\mathbf{V}(k)$  are assumed to be independent, so that

$$\mathbf{E}\{\mathbf{V}(j)\mathbf{W}(k)\} = \mathbf{0} \quad (52)$$

for all  $j = 1, 2, \dots$ , and  $k = 0, 1, 2, \dots$ . The intial state  $\mathbf{X}(0)$  is assumed to be a gaussian random vector with

$$\mathbf{E}\{\mathbf{X}(0)\} = \mathbf{0} \quad (53)$$

It is assumed that  $\mathbf{X}(0)$  is independent of  $\mathbf{W}(k)$  and  $\mathbf{V}(k)$ .

The optimal estimate of  $\mathbf{X}(k + 1)$  is denoted  $\hat{\mathbf{X}}(k + 1 | k + 1)$ . The Kalman filter is designed to minimize

$$\mathbf{J} = \mathbf{E}\{[\mathbf{X}(k + 1) - \hat{\mathbf{X}}(k + 1 | k + 1)]^T[\mathbf{X}(k + 1) - \hat{\mathbf{X}}(k + 1 | k + 1)]\} \quad (54)$$

The recursive realtions for generating  $\hat{\mathbf{X}}(k + 1 | k + 1)$  are

$$\hat{\mathbf{X}}(k + 1 | k) = \mathbf{A}\hat{\mathbf{X}}(k | k) + \mathbf{B}\mathbf{U}(k) \quad (55)$$

$$\hat{\mathbf{X}}(k + 1 | k + 1) = \hat{\mathbf{X}}(k + 1 | k) + \mathbf{G}(k + 1)[\mathbf{Y}(k + 1) - \mathbf{C}\hat{\mathbf{X}}(k + 1 | k)] \quad (56)$$

for  $k = 0, 1, 2, \dots$ , where  $\hat{\mathbf{X}}(0 | 0) = \mathbf{0}$ .  $\hat{\mathbf{X}}(0 | 0)$  is set equal to zero since the expectation of  $\mathbf{X}(0)$  is zero.

$\mathbf{G}(k + 1)$  is an  $n \times m$  matrix, called the Kalman Gain Matrix which is specified by the realtions:

$$\mathbf{P}(k + 1 | k) = \mathbf{A}\mathbf{P}(k | k)\mathbf{A}^T + \mathbf{B}\mathbf{Q}(k)\mathbf{B}^T \quad (57)$$

$$\mathbf{G}(k + 1) = \mathbf{P}(k + 1 | k)\mathbf{C}^T[\mathbf{C}\mathbf{P}(k + 1 | k)\mathbf{C}^T + \mathbf{R}(k + 1)]^{-1} \quad (58)$$

$$\mathbf{P}(k + 1 | k + 1) = [\mathbf{I} - \mathbf{G}(k + 1)\mathbf{C}]\mathbf{P}(k + 1 | k) \quad (59)$$

$P(k | k)$  is the covariance matrix of the error between the states and their estimates

$$P(k | k) = E\{[X(k) - \hat{X}(k | k)][X(k) - \hat{X}(k | k)]^T\} \quad (60)$$

Since we are using the steady state gains the choice of  $P(0 | 0)$  is irrelevant.  $P(0 | 0)$  is initialized to zero in the gain derivation program for simplicity [Ref. 6: p. 139-140].

### C. STEADY-STATE SOLUTION

If Equations (57), (58), and (59) are repeatedly iterated,  $G(k + 1)$  will converge to a steady state value [Ref. 7: p. 263].

$$G_{ss} = \lim_{k \rightarrow \infty} G(k + 1) \quad (61)$$

The values of  $G_{ss}$  (or  $G$ ) can be substituted into Eq. (56) making the steady state Kalman filter

$$\hat{X}(k + 1 | k) = A\hat{X}(k | k) + BU(k) \quad (62)$$

$$\hat{X}(k + 1 | k + 1) = \hat{X}(k + 1 | k) + G[Y(k + 1) - C\hat{X}(k + 1 | k)] \quad (63)$$

### D. OBSERVER PERFORMANCE

The performance of an observer is judged by how accurately and rapidly it estimates the desired states. The performance measure of the observer as a whole is shown in equation (54). The normalized performance of the observer for individual states is

$$J_i = E[(x_i - \hat{x}_i)^2] / E[x_i^2] \quad (64)$$

which can be found using Eq. (65)

$$J_i = \sum_{k=0}^{\infty} (x_i(k) - \hat{x}_i(k))^2 T_s \div \sum_{k=0}^{\infty} x_i^2(k) T_s \quad (65)$$

$J_i$  = performance measure for the  $i$ th state

$x_i(k)$  = value of the  $i$ th state at  $k$

$\hat{x}_i(k)$  = observer estimate of  $i$ th state at  $k$

$T_s$  = sample interval

A normalized performance measure is used to aid comparison of the performance of the observer in estimating various states. From Eq. (65) it can be shown that if  $\hat{x}_i(k) = 0$  for all  $k = 0, 1, 2, \dots$  that  $J_i$  would be unity. Therefore, the better the performance of the observer, the smaller the fraction of one  $J_i$  will be.

## IV. SIMULATION

### A. INTRODUCTION

The objectives of the simulation were to

- determine the sensitivity of the observer performance and settling time to changes in the ratio of plant noise to measurement noise,
- determine the effect on observer performance and settling time of increasing the number of modes observed in the matched plant/observer, and
- determine the performance for the reduced order observer.

### B. PLANT AND OBSERVER DATA

The dynamic model is a truncated form of a preliminary space station configuration; the phase II dual keel structure.<sup>1</sup> The full model consists of an infinite number of natural modes but this was restricted to the first ten active modes for this study due to limitations on computer resources. As will be shown reasonable data can be obtained with this simplification in examining the observer performance.

### C. SIMULATION PROGRAMS

The simulation was broken down into two segments due to the large memory and computational time requirements. The first program computed the steady state observer gain matrix ( $G$ ). The second program ran the observer and the plant when the plant was subjected to an impulse excitation.

The steady state observer gain matrix ( $G$ ) was obtained by repeated iteration of equations (57), (58), and (59). The equations were run until the values of the matrix changed by less than a set fraction. The following formula was used to check the changes in the gain matrix elements

$$\Delta g_{i,j} = [g_{i,j}(k+1) - g_{i,j}(k)]/g_{i,j}(k+1) \quad (66)$$

The program was terminated when  $\Delta g_{i,j}$  was less than  $10^{-10}$ .

The settling time for the estimates of the states to be within 2% of the actual states was determined by finding the eigenvalues of  $A - G^*C$  then computing as follows [Ref. 6: p. 139-143]

---

<sup>1</sup> The model for preliminary station configuration was provided courtesy of McDonnell Douglas Astronautics Company, 5301 Bolsa Avenue, Huntington Beach, CA 92647.

$$T_s = \log(.02) / \log(\lambda_{AGC_{min}}) \quad (67)$$

The expected error in the sensor, i.e., the standard deviation of the noise in the measurement, was chosen as  $10^{-3}$  feet per sec. per sec. based on the natural frequencies in the structure and reasonable sensor sensitivity [Ref. 2: p. 79-80]. The expected plant noise was varied to find the range of ratios between plant and measurement noise that the filter would be effective. This approach was taken since the plant noise contributors are not currently well defined.

The second program subjected the plant as modeled in Eq. (45) to an impulse excitation and then had the observer estimate the selected states using observer equations (62) and (63). Observer performance was computed using Eq. (65).

A third program was used to find the contribution of unobserved modes to the noise in the Kalman observer. The program ran the plant subject to an impulse excitation and computed the product of the measurement matrix  $C$  times the unobserved modes of the state vector  $X(k)$  for a measure of the noise contributed by the unobserved modes.

The three programs are listed in the appendices.

#### D. EFFECT OF PLANT TO MEASUREMENT NOISE RATIO ON OBSERVER PERFORMANCE

The ratio of the variance of the plant noise ( $P_N$ ) to the variance of the measurement noise ( $M_N$ ) was found to have a strong effect on the Kalman Observer performance ( $J$ ) and settling time ( $T_s$ ). Figures 1 through 6 show the observer performance for a 3 mode matched plant and observer system for progressively smaller  $P_N/M_N$  ratios.<sup>2</sup> Figure 7 shows the performance for the seventh mode (position) versus several values of  $P_N/M_N$ . Figure 8 is the settling time versus the same  $P_N/M_N$  ratios.

The figures show that, for all of the plotted performance values, the observer performance is at least marginally acceptable regardless of the  $P_N/M_N$  ratio. Decreasing the  $P_N/M_N$  ratio leads to an even more rapid degradation in observer performance. The settling times also rapidly increase as the  $P_N/M_N$  ratio decreases.

---

<sup>2</sup> Figures 1-6 and 9-15 show the performance measure for each mode. The bar for the mode position is immediately to the right of the numbered tick mark on the x-axis scale, the mode velocity is next to it immediately adjacent to the tick mark without a number.

## E. EFFECTS OF INCREASED MODES ON OBSERVER PERFORMANCE

The matched plant/observer was run with increasing numbers of modes to see if there was an effect on observer performance ( $J$ ) or settling time  $T$ . Figures 7 through 17 are of observer performance for systems with increasing numbers of modes in the system being observed. Figure 18 is of settling time versus the number of modes in the system. The ratio of  $PN/MN$  was kept constant at  $PN/MN = 2.5 \times 10^9$ .

The increasing of the number of modes for the matched plant/observer had negligible effect on the performance for the individual modes. The performance value for the modes was effectively constant. Settling times for the observers increased as the number of modes was increased.

## F. REDUCED ORDER KALMAN OBSERVER

The Kalman Observer has been shown to be effective where the number of modes observed matches the number of modes in the plant. The Kalman Observer was then run with the one less mode observed than the number of modes in the plant. The gain matrix ( $G$ ) from the matched system was used. The observer failed with the state estimates produced by the observer becoming excessively large and having settling times of hours vice minutes. Since the purpose of the observer was to provide estimates for use in controlling the plant the time delay makes the estimates unusable.

The cause of the observer failure is apparent when you look at the last portion of Eq. (56) of the Kalman Observer

$$G[Y(k+1) - C\hat{X}(k+1|k)] \quad (68)$$

This portion of the observer equation is the correction of  $\hat{X}(k+1|k)$  to produce  $\hat{X}(k+1|k+1)$ . The design of the Kalman observer is to produce an estimate despite the measurement noise but, with the reduced order filter there is additional unanticipated noise which causes over correction of the values of  $\hat{X}$  leading to the state estimates being excessively large and settling times being too long. This can be shown by examining what composes  $Y(k+1) - CX(k+1|k)$

$$\mathbf{C} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \uparrow \\ \downarrow \\ x_{m-1}(k) \\ x_m(k) \\ \cdots \\ x_{m+1}(k) \\ x_{m+2}(k) \\ \uparrow \\ \downarrow \\ x_{n-1}(k) \\ x_n(k) \end{bmatrix} - \mathbf{C} \begin{bmatrix} \hat{x}_1(k) \\ \hat{x}_2(k) \\ \uparrow \\ \downarrow \\ \hat{x}_{m-1}(k) \\ x_m(k) \\ \cdots \\ 0 \\ \uparrow \\ \downarrow \\ 0 \end{bmatrix} = 0 \quad (69)$$

$\mathbf{C}$  times the state  $x_{m+1}(k)$  through  $x_n(k)$  is unanticipated noise so if

$$\mathbf{C} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \uparrow \\ \downarrow \\ x_{m-1}(k) \\ x_m(k) \end{bmatrix} - \mathbf{C} \begin{bmatrix} \hat{x}_1(k) \\ \hat{x}_2(k) \\ \uparrow \\ \downarrow \\ \hat{x}_{m-1}(k) \\ \hat{x}_m(k) \end{bmatrix} = 0 \quad (70)$$

the remaining portion of the  $\mathbf{C}$  matrix times the modal states is an equivalent noise.

Table (1) shows the growth of the unanticipated noise in the filter as the number of unobserved modes in the plant grows. Table (2) shows the individual contributions of the individual modes when left unobserved. Table (1) shows that the unanticipated noise is much larger than that expected by the filter ( $10^{-3}$ ). Table (2) shows that there are modes that do not markedly contribute to the noise and that they might successfully be left unobserved if the measurement noise estimate was already much larger than these noise sources.

**Table 1. CUMULATIVE UNANTICIPATED NOISE FROM UNOBSERVED MODES**

Number of Unobserved Modes	Unobserved Modes	E1	E2	E2
1	10	0.647	97.440	3.277
2	10-11	366.354	95.764	3.142
3	10-12	366.355	95.764	3.142
4	10-13	365.565	95.855	3.143
5	10-14	365.426	96.032	5.704
6	10-15	144195.7	116.205	2201.94
7	10-16	148006.8	170.142	5475.21
8	10-17	473974.3	39424.1	5692.50
9	10-18	474344.2	60078.8	9419.77
10	10-19	474358.5	68987.7	9865.27

**Table 2. UNANTICIPATED NOISE FROM UNOBSERVED MODES BY MODE**

Unobserved Mode	E1	E2	E3
10	0.64716	97.4403	3.27761
11	359.342	0.20929	0.21429
12	0.9353E-08	0.1364E-07	0.2196E-07
13	0.3852E-02	0.4409E-02	0.9017E-03
14	0.5615E-03	0.2592E-01	2.57554
15	143090.9	17.9682	2167.02
16	195.736	83.3458	5675.91
17	324471.2	39252.26	221.170
18	216.240	21133.6	3736.93
19	7.69298	8829.95	458.504
20	2.3194	3949.16	108.981

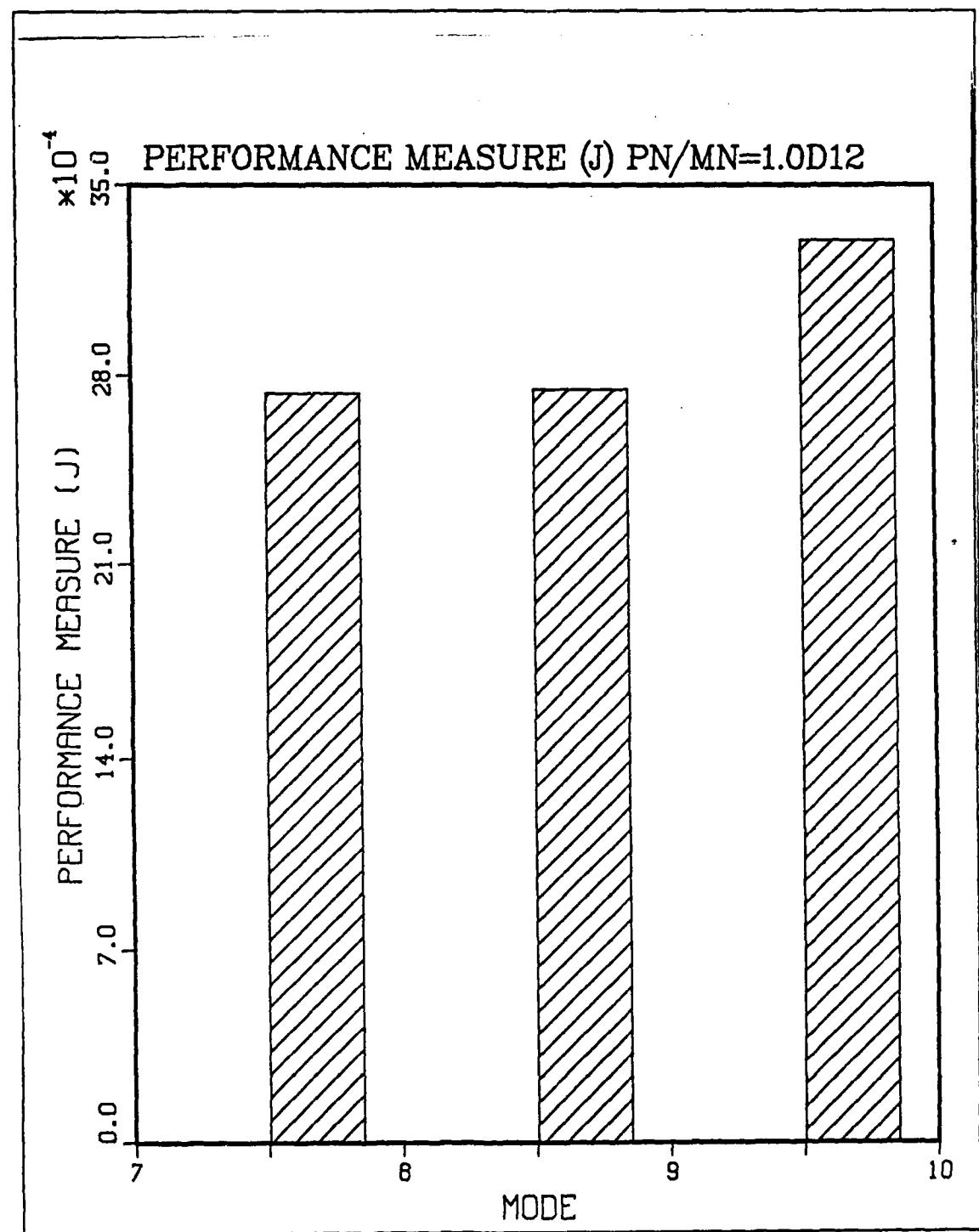


Figure 1. Observer Performance (J)  $PN/MN = 1.0d12$

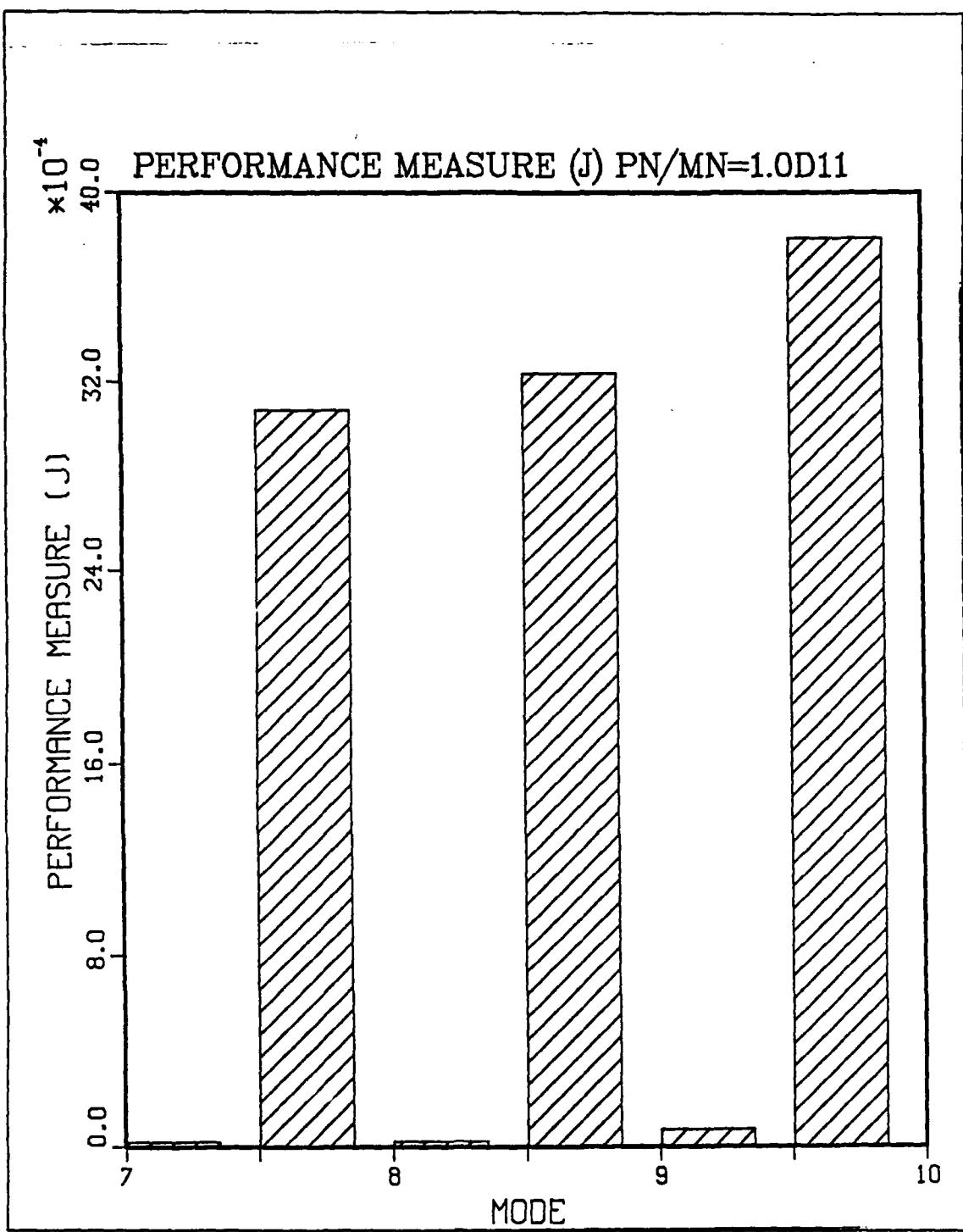


Figure 2. Observer Performance (J) PN/MN = 1.0d11

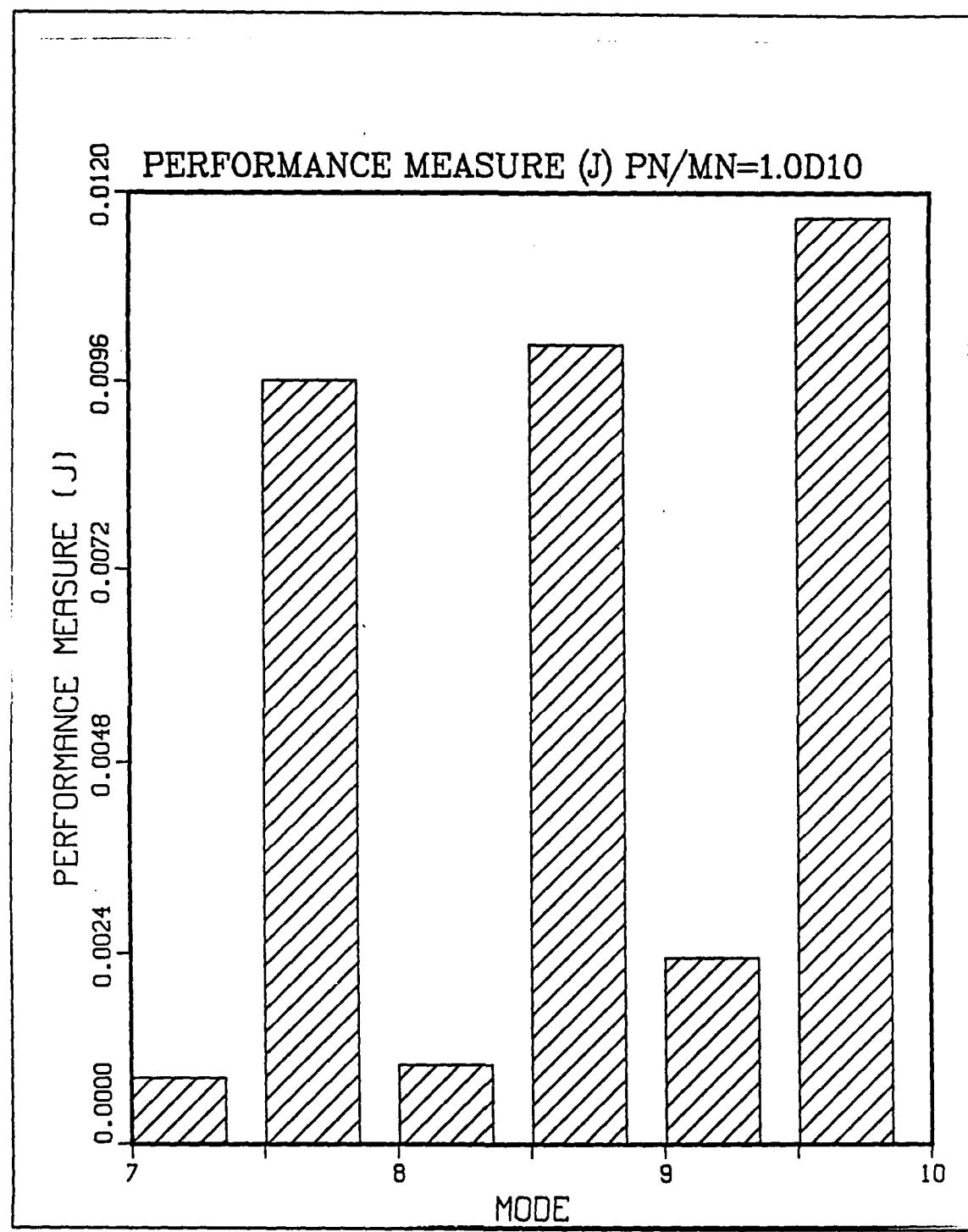


Figure 3. Observer Performance (J) PN/MN = 1.0d10

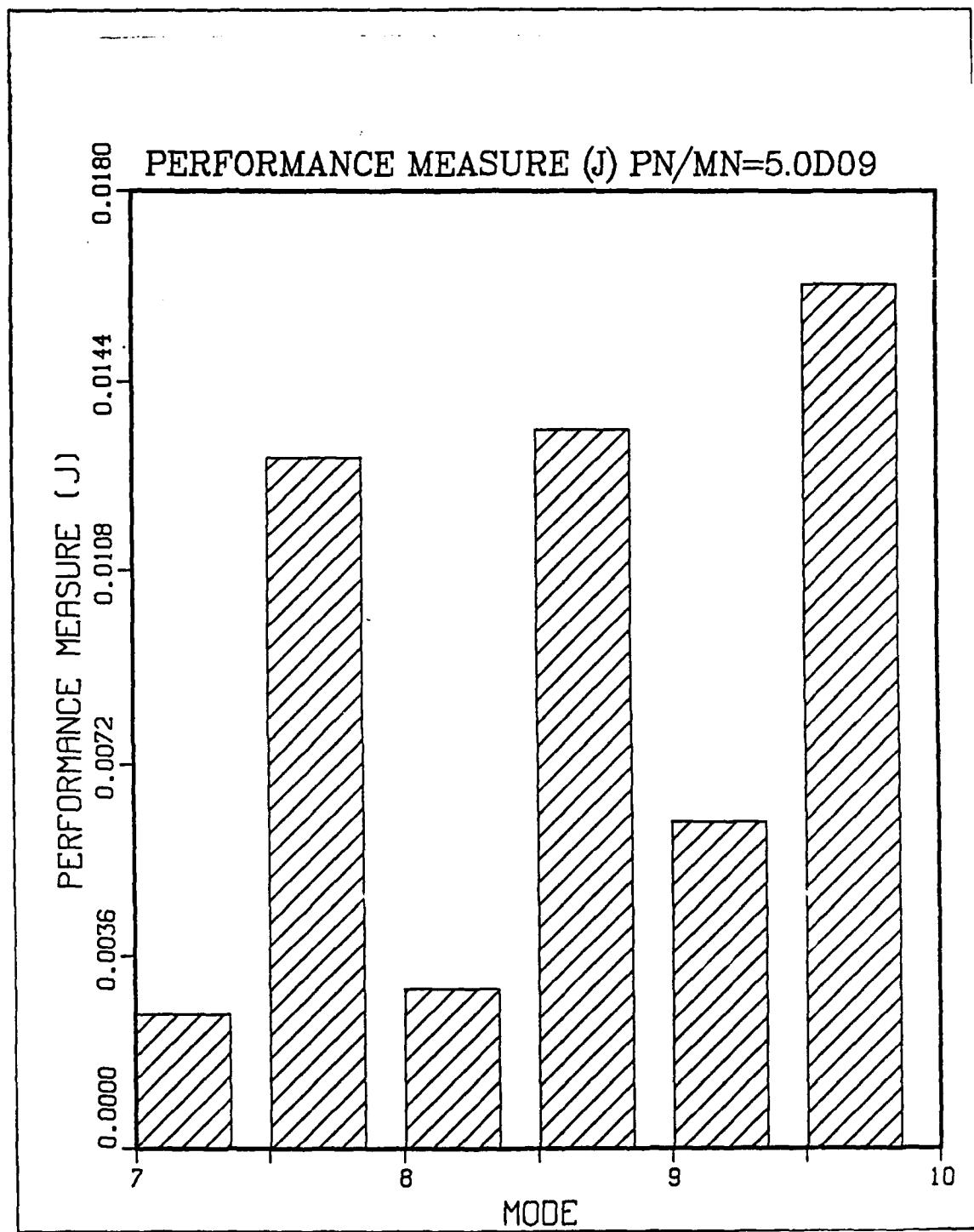


Figure 4. Observer Performance (J) PN/MN = 5.0d09

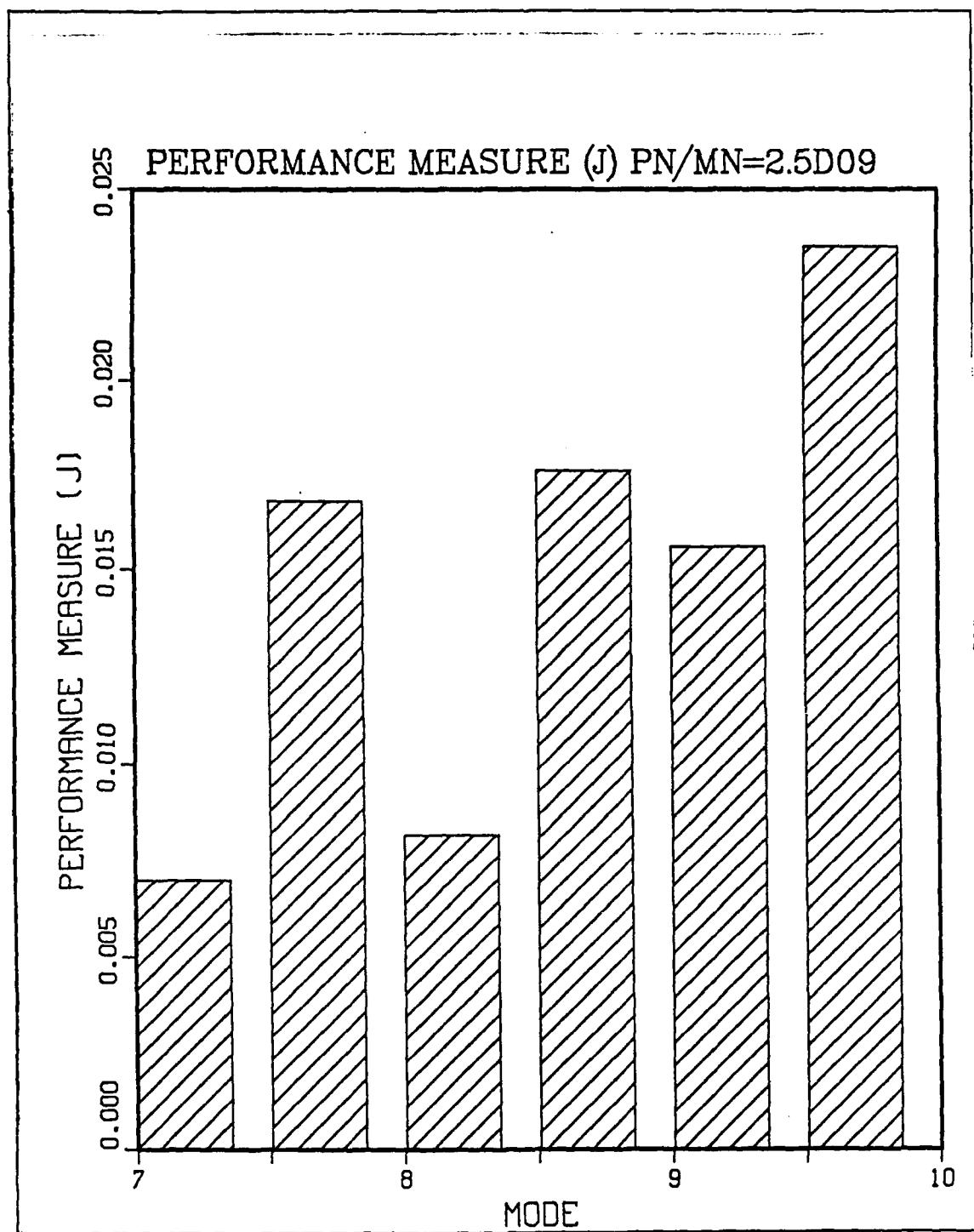


Figure 5. Observer Performance (J) PN/MN = 2.5d09

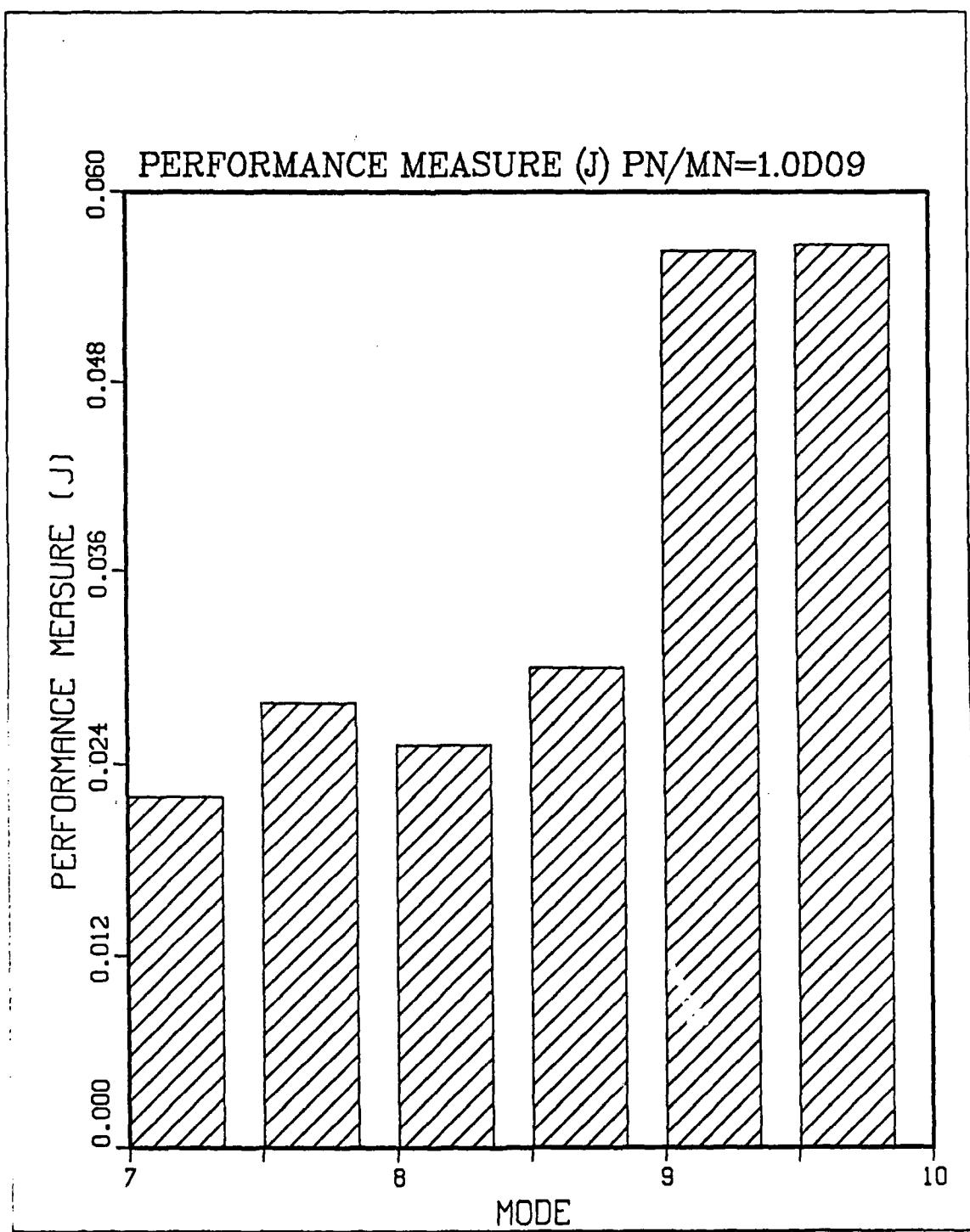


Figure 6. Observer Performance (J) PN/MN = 1.0d09

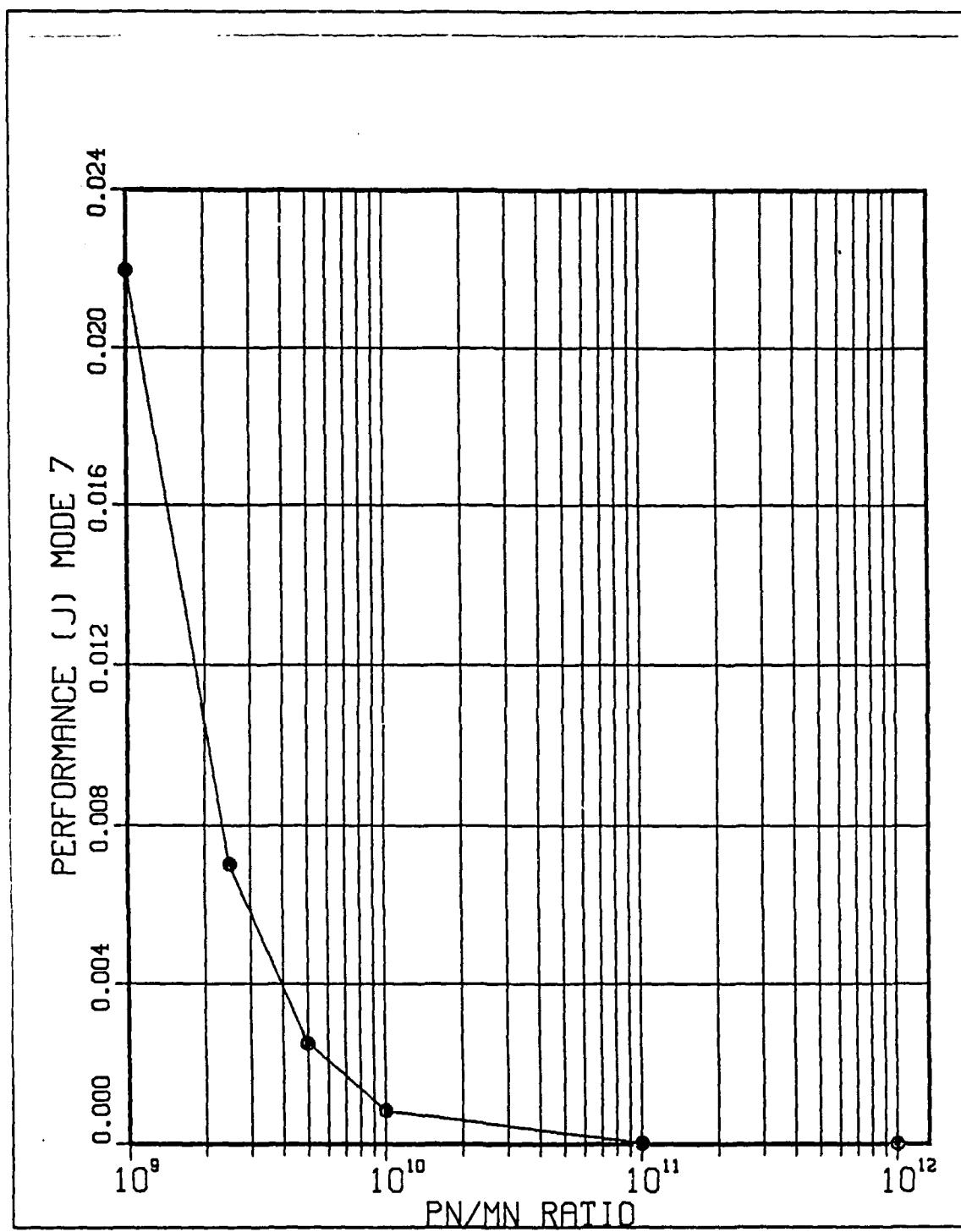


Figure 7. Mode 7 (Position) Observer Performance versus PN/MN

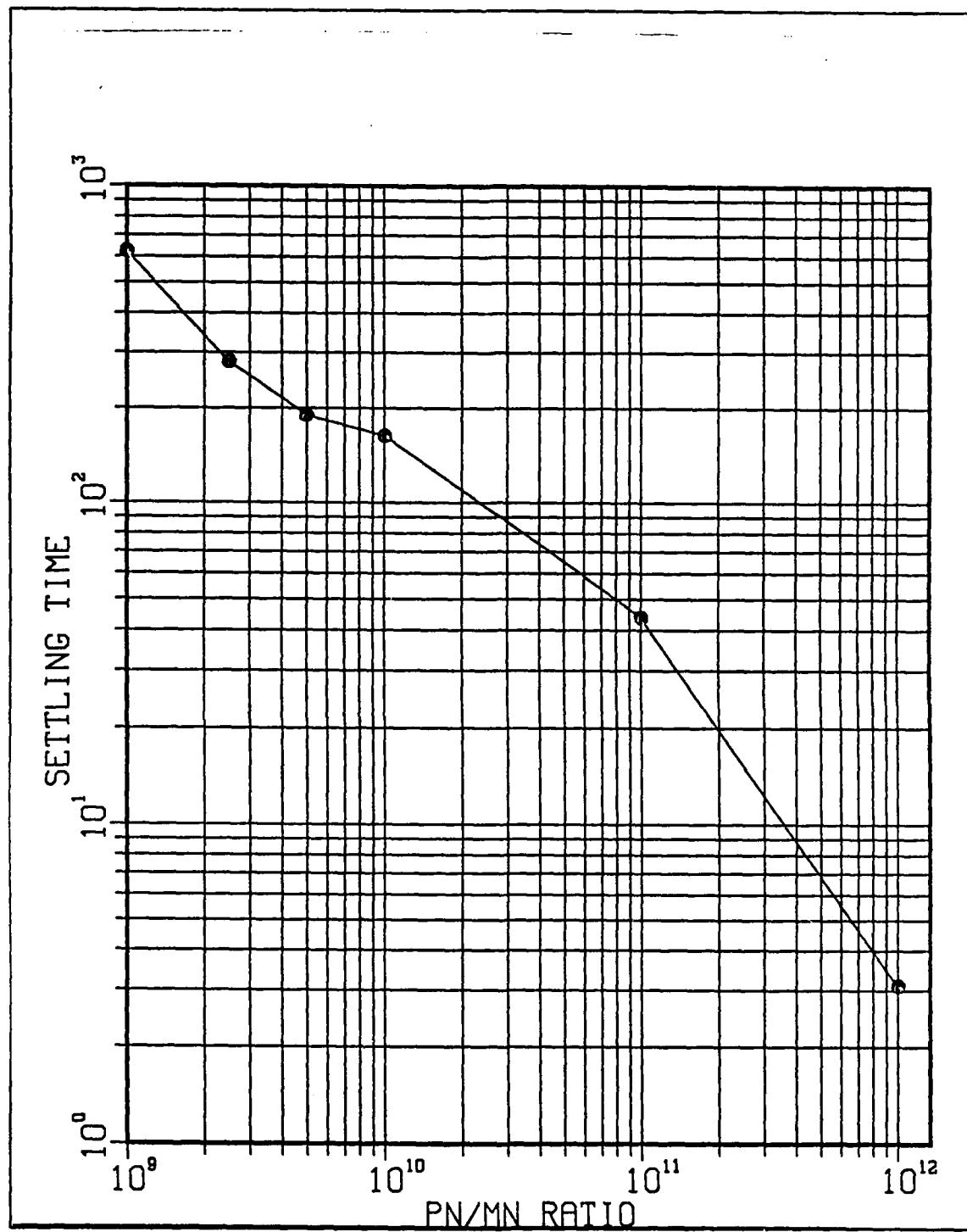


Figure 8. Settling Time versus PN/MN

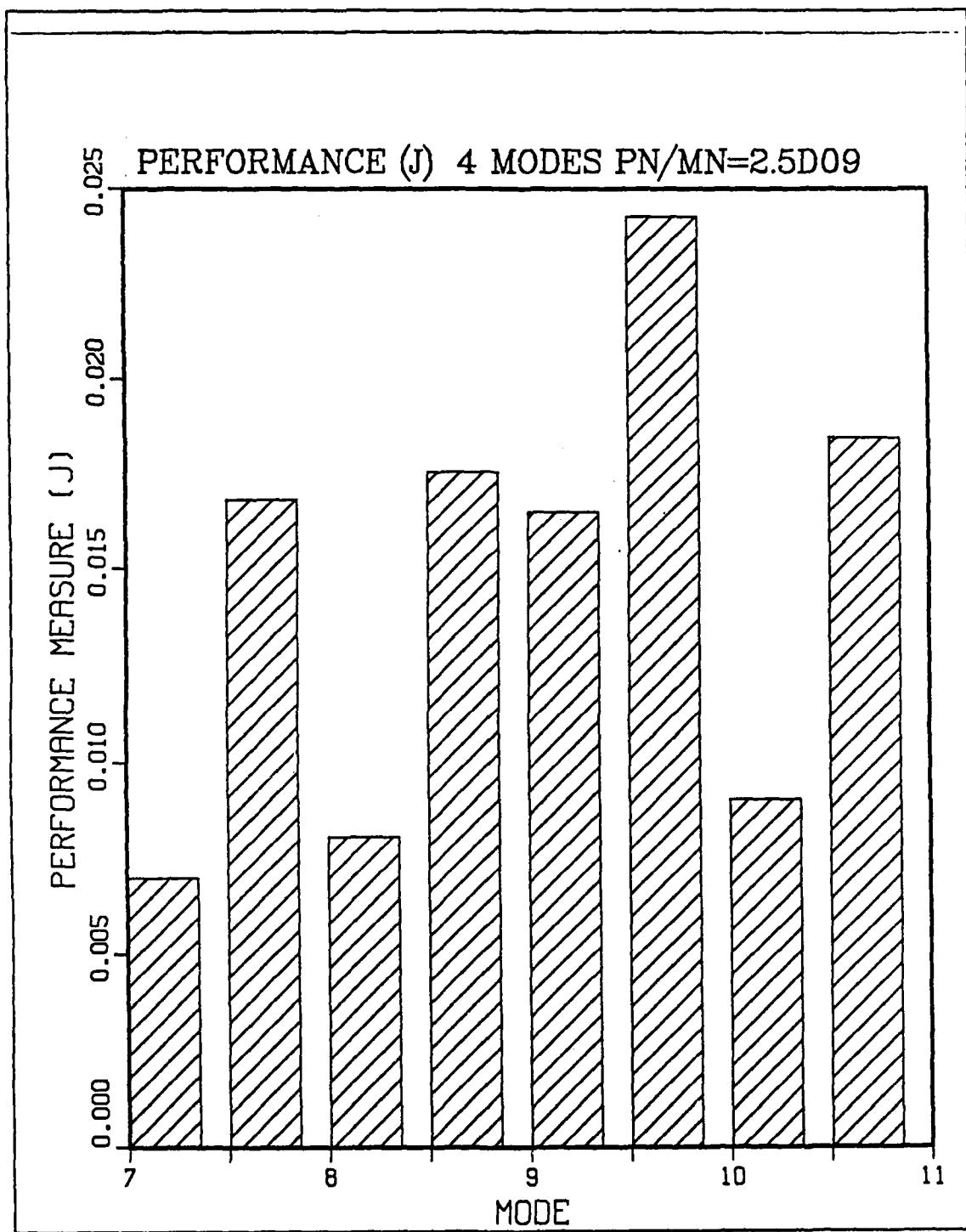


Figure 9. Observer Performance (J) 4 Modes (7 - 10)

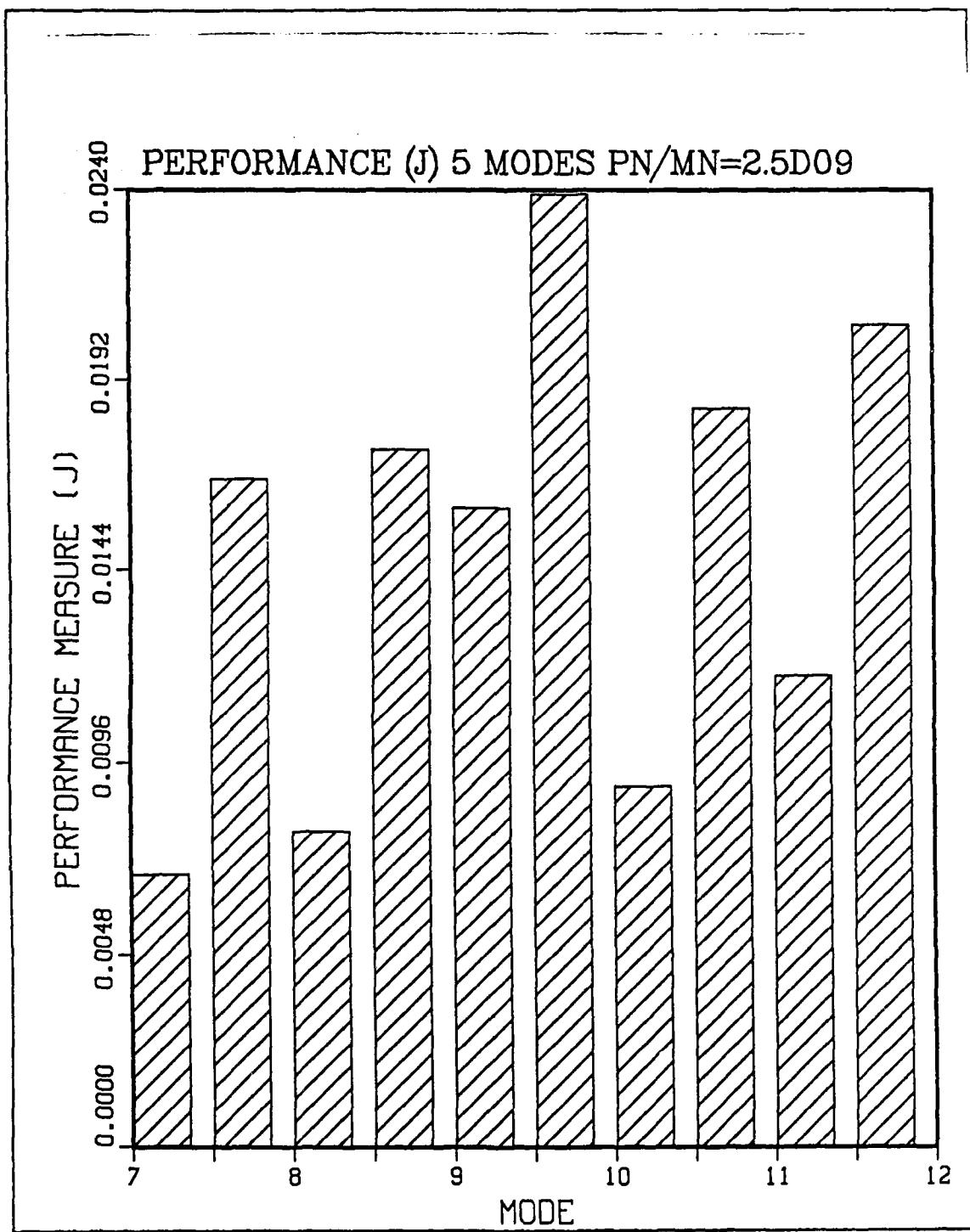


Figure 10. Observer Performance (J) 5 Modes (7 ~ 11)

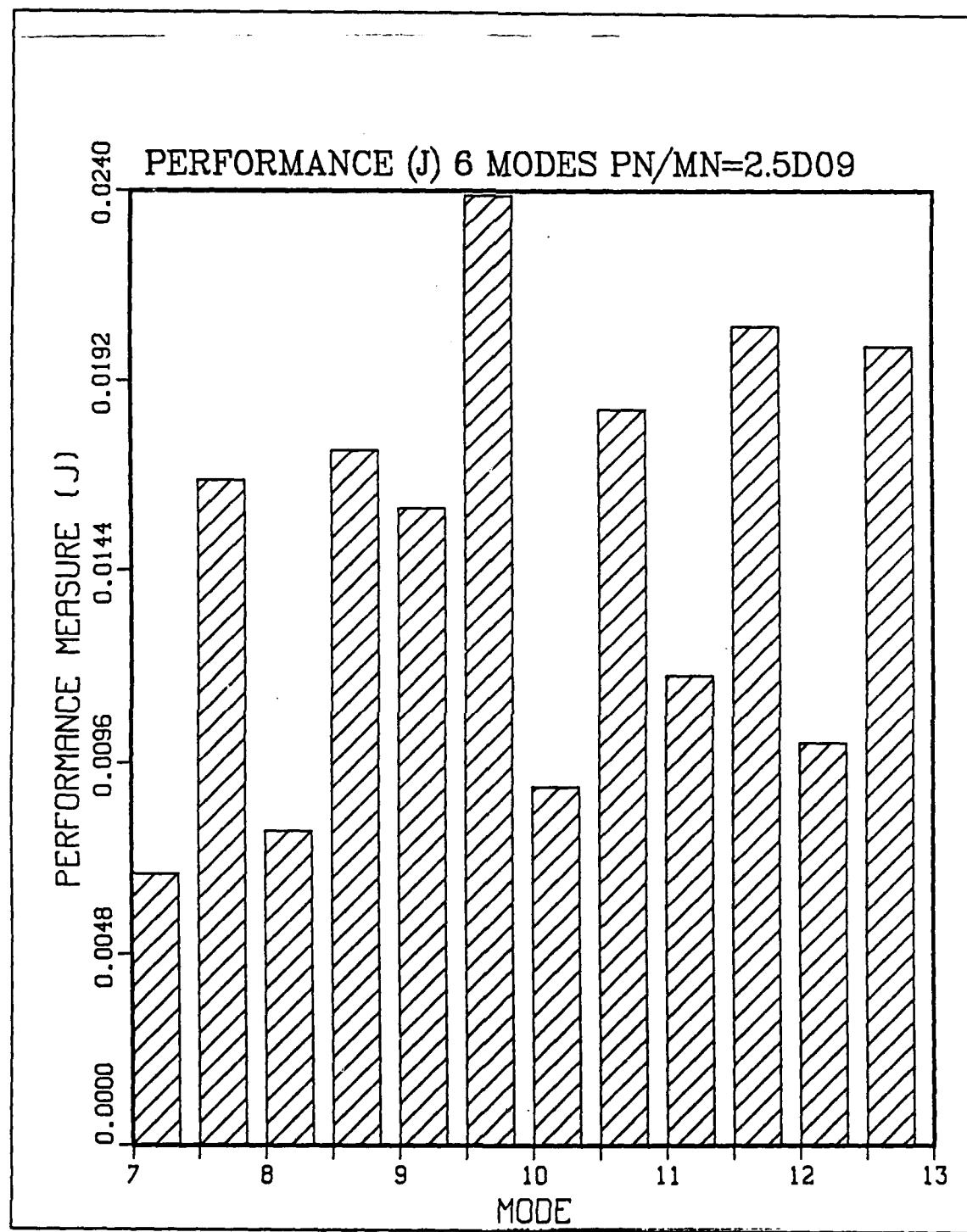


Figure 11. Observer Performance (J) 6 Modes (7 - 12)

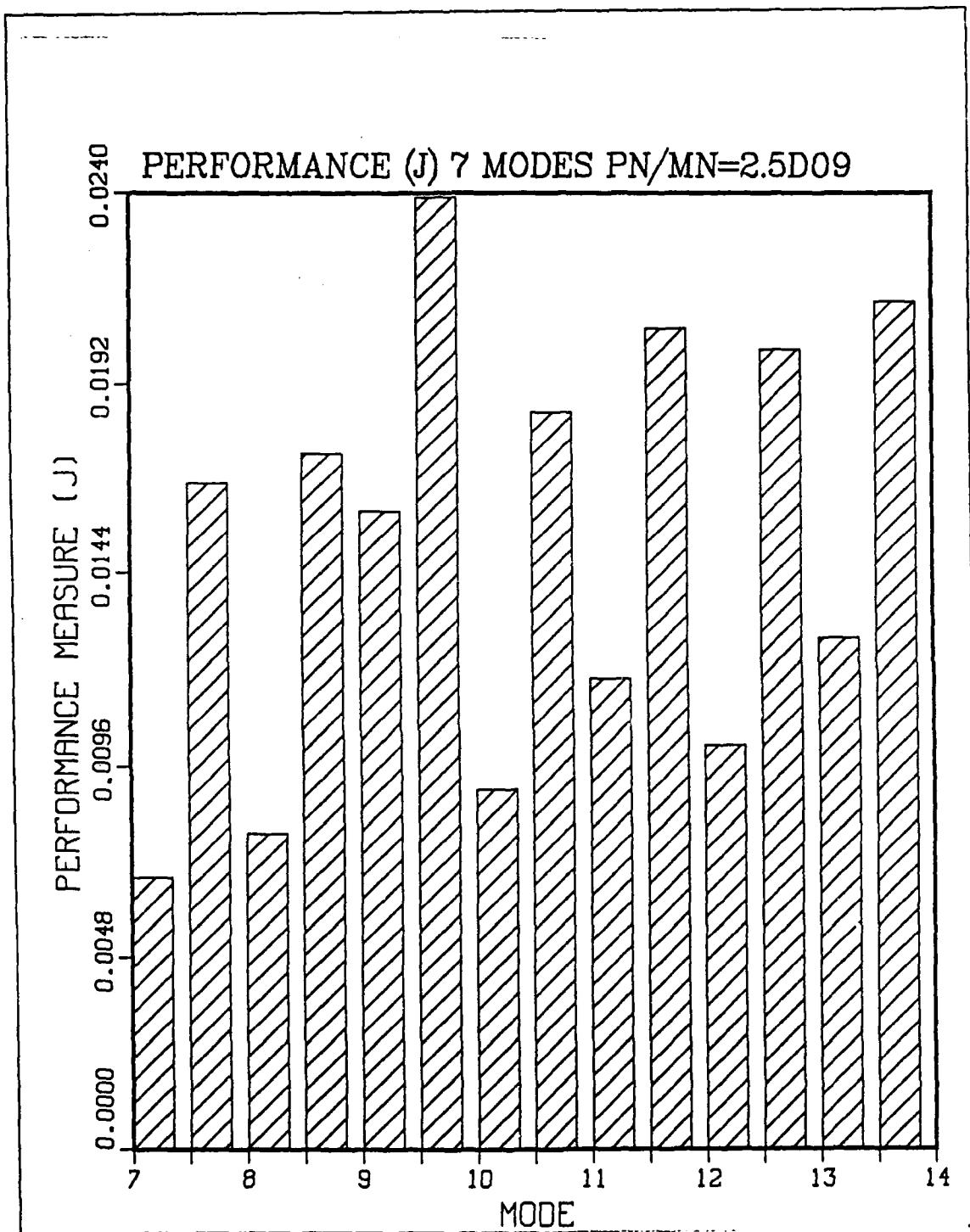


Figure 12. Observer Performance (J) 7 Modes (7 - 13)

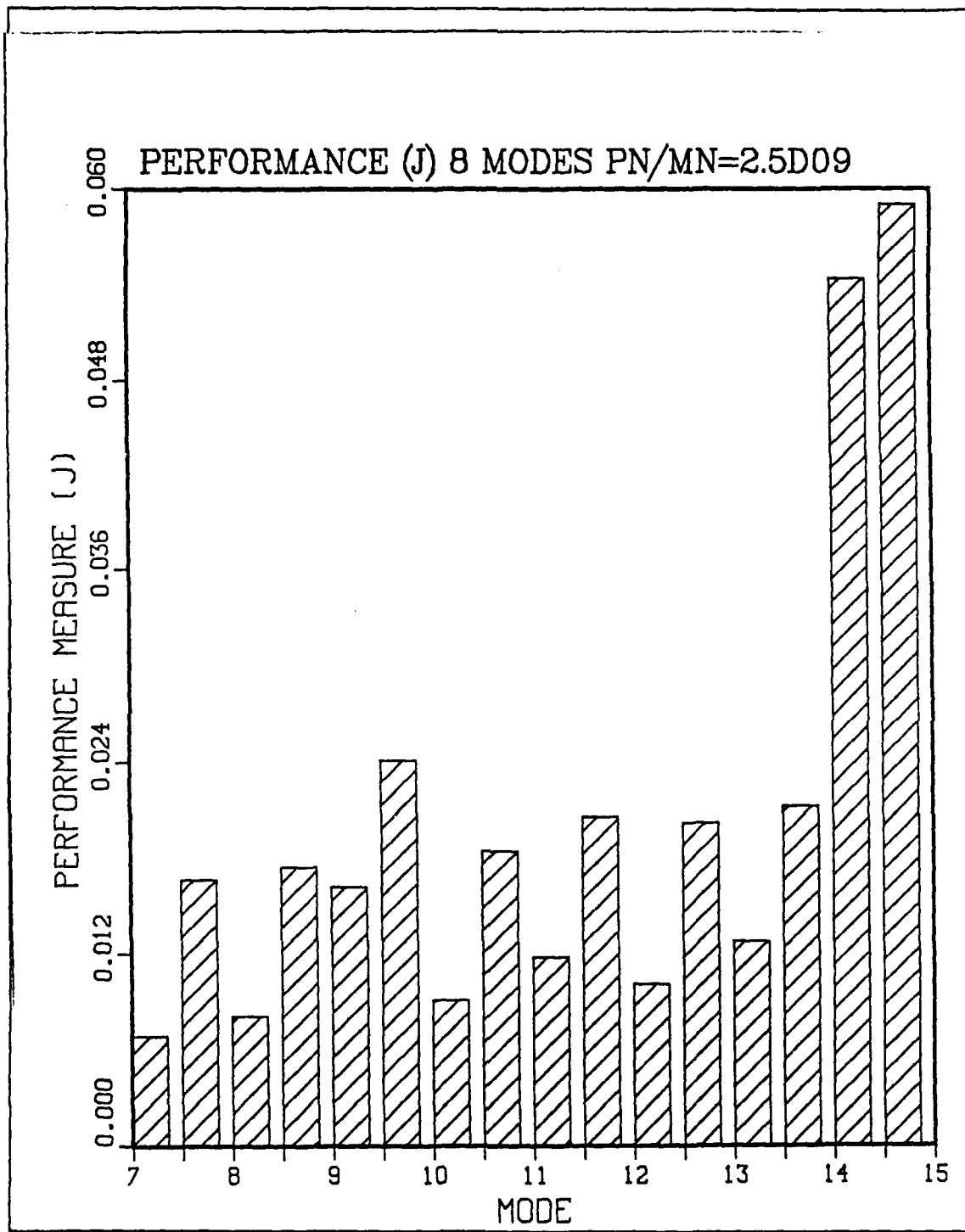


Figure 13. Observer Performance (J) 8 Modes (7 - 14)

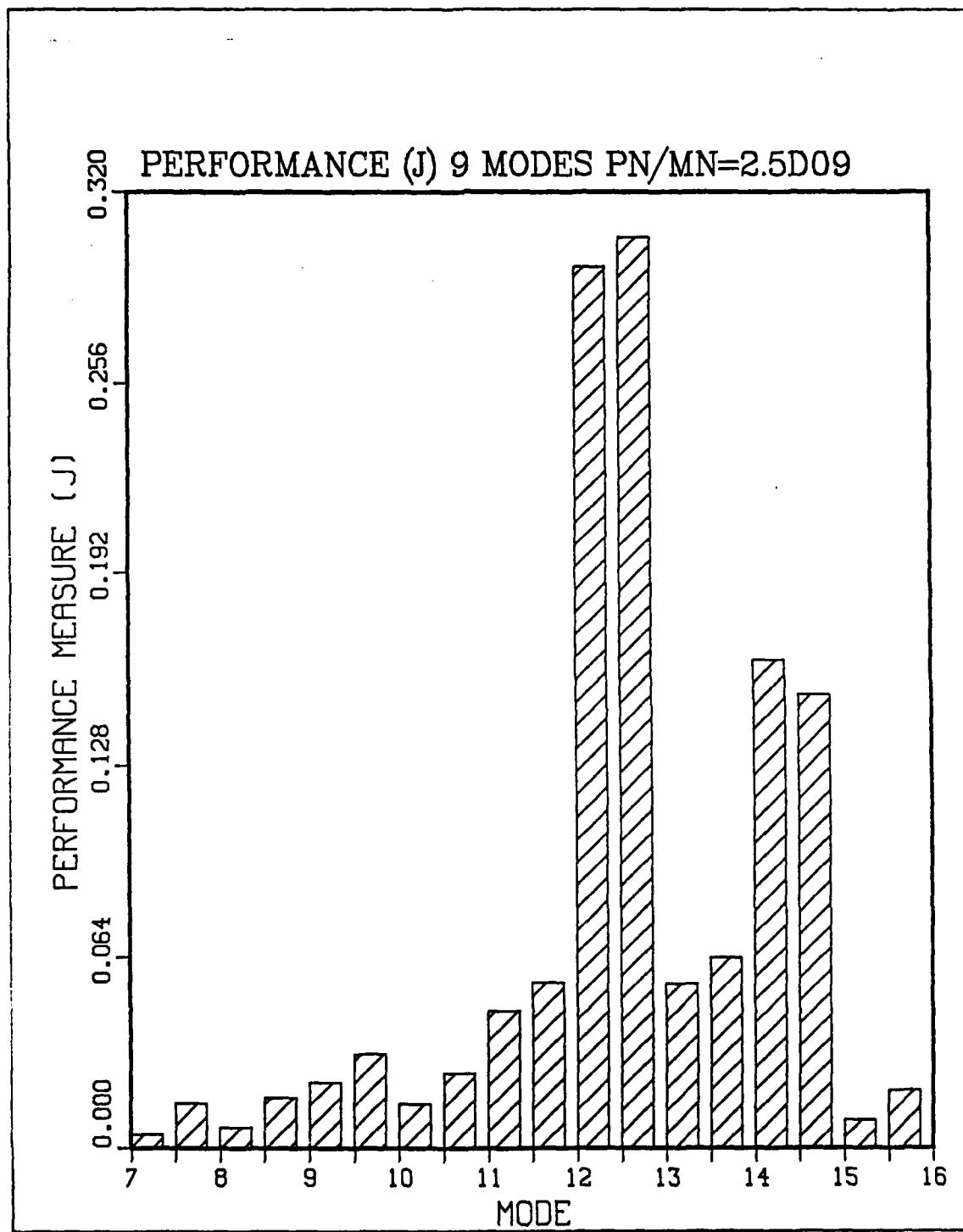


Figure 14. Observer Performance (J) 9 Modes (7 - 15)

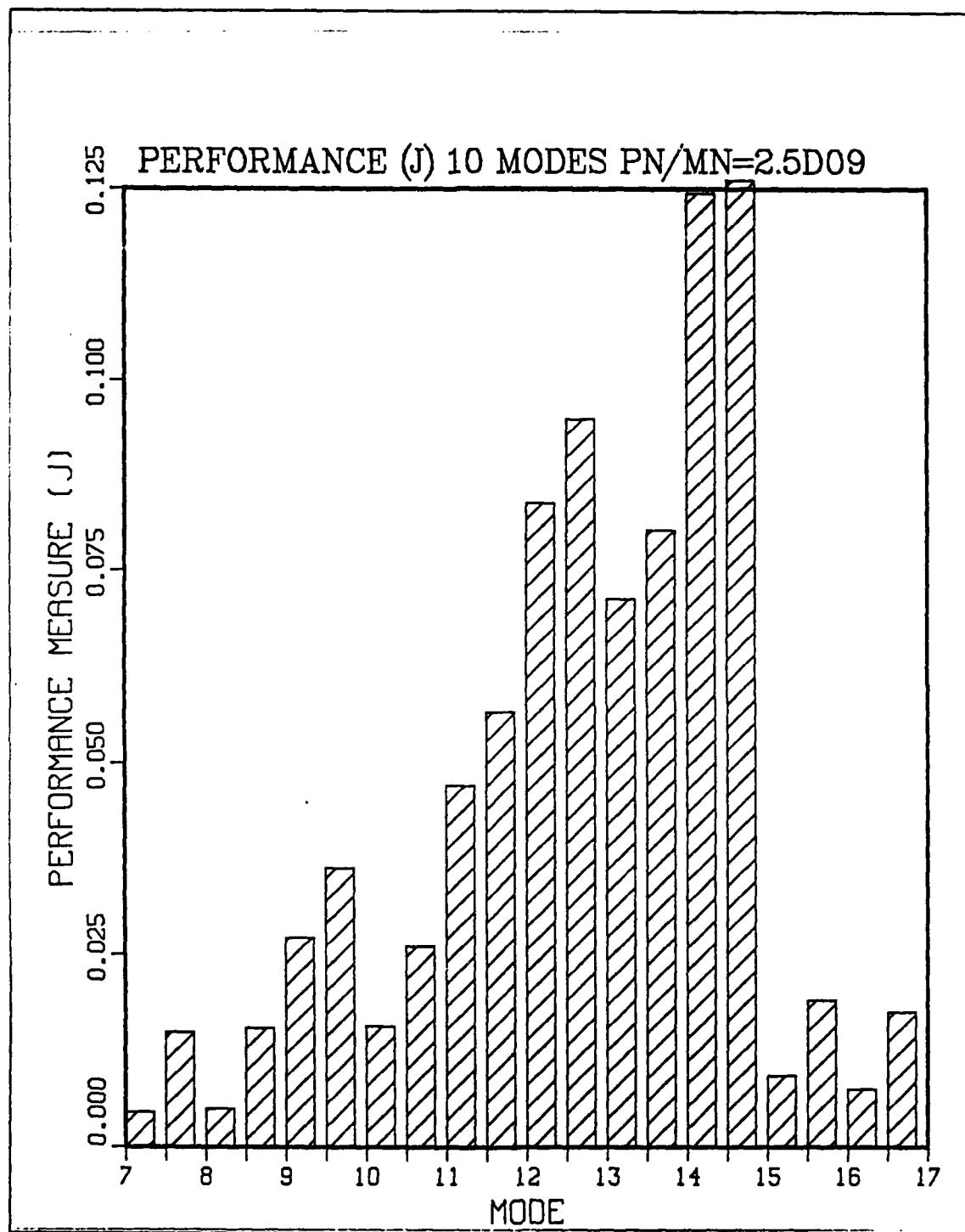


Figure 15. Observer Performance (J) 10 Modes (7 - 16)

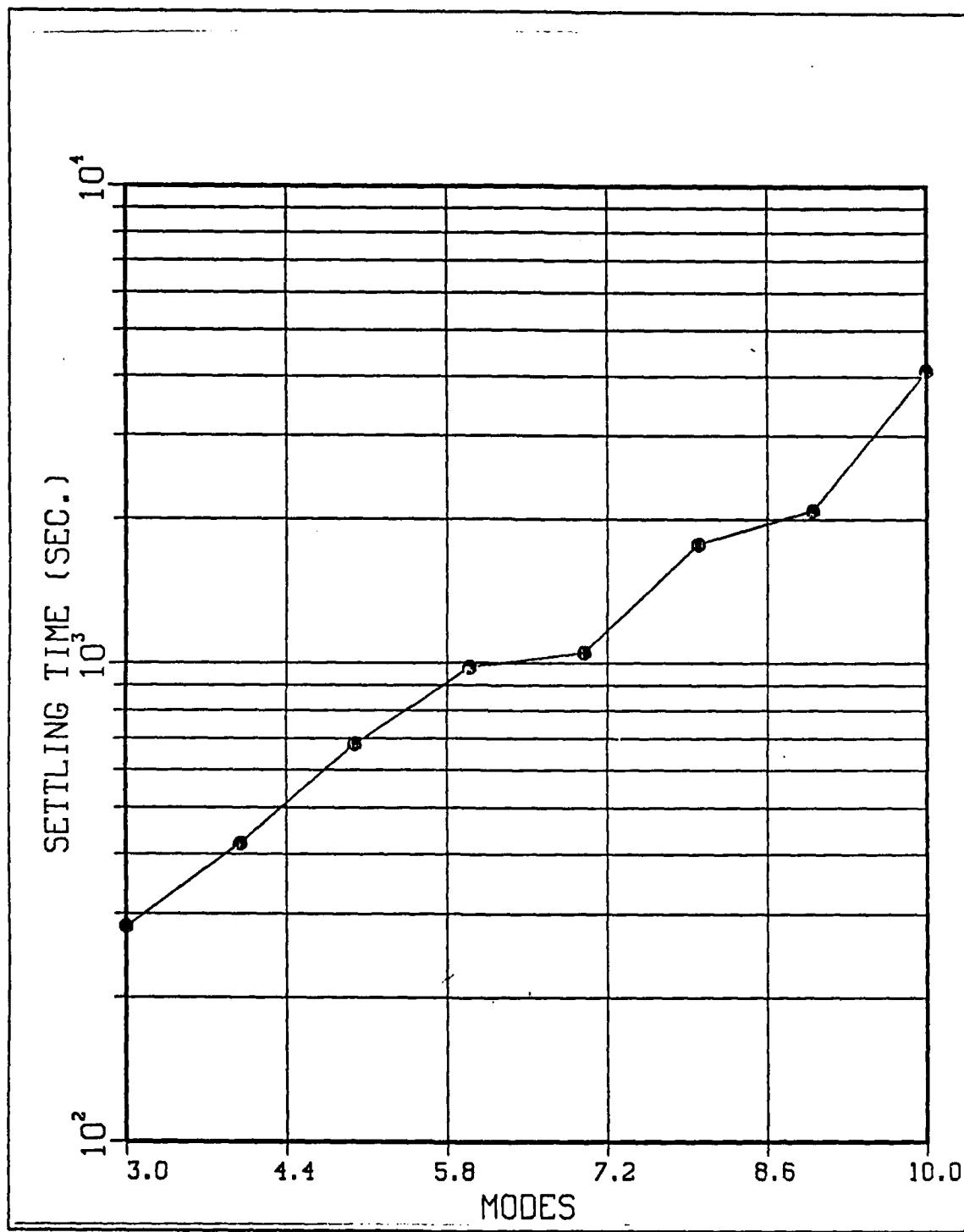


Figure 16. Settling Time versus number of Modes Observed

## V. CONCLUSIONS AND RECOMMENDATIONS

### A. CONCLUSIONS

Simulations runs showed that a matched plant/observer can work if the following criterions are meet:

- The ratio of plant noise to measurement noise is sufficiently high to produce a usable settling time.
- Sufficient computational power is available to run the matched observer. The amount of memory and number of computation goes up as the number of modes observed increases.

Utilizing a reduce order observer for an arbitrarily selected set of modes is not feasible. The non-observed modes add so much noise to the system that settling times and observer performance are so poor as to render the observer useless for obtaining state values for plant control.

### B. RECOMENDATIONS

The work on the Kalman Observer for Large Space Structures lead to the following recommendations for further research:

- Identify those modes that contribute the largest noise to the Kalman observer and set the observer to estimate these states in addition to those required for plant control. A possible method for identifying the modes that contribute the largest noise to the observer would be the Karhunen-Loeve expansion.
- Modifying the plant/observer to model the use of sensors at additional positions to see if the increase in the data rate will help decrease settling time.
- Modify the model to incorporate noise injection into more than one location. The current model has noise injected at only one position, a useful simplification for initial analysis but not realistic.

## APPENDIX A. KALMAN GAIN MATRIX GENERATION PROGRAM

```

***** GAIN *****  

***** ADAPTED TO RUN KALMAN FILTER AND COMPUTE THE *****  

***** G MATRIX BY ITERATION STOPPING WHEN THE *****  

***** THE MATRIX GOES TO STEADY STATE *****  

***** VARIABLE DECLARATIONS *****  

EXTERNAL STMTRX,DLINRG,EXCMS, DEVCRG  

CHARACTER*6 NAM  

CHARACTER*1 AGAIN, CORECT, RAGAIN  

INTEGER ROWN1, ROWN2, ROWN3, COUNT, NODE, MODE, KQ, EMODE, SMODE, R2M, C2M  

INTEGER CT, CF, KADJ, CFADJ, LOOP, PRNT, JJ, JK, N1, JR, KR, MR, ISEED, M2  

INTEGER JL, J1, JM  

REAL LAMA(100), UGVEX(684,100), RNODE, RMODE, MIN  

REAL*8 PHI(2,2,100), GAMMA(2,100), EGT, GMA, WN, W1, X1T, X2T, TIME  

REAL*8 A(200,200), B(200,3), F(3, 50), IMPLSE, ENERGY  

REAL*8 COSW1T, SINW1T, X1(100), X2(100), COST(100)  

REAL*8 DAMP, SAMPT, PI, SAMPTM, SUM1, SUM2, SUM3, SUMC  

REAL*8 C(6,200), IDENT( 50, 50), RMN(6,6), QPN(3,3)  

REAL*8 PK( 50, 50), Y(6), BN(200,3)  

REAL*8 PNVARX, PNVARY, PNVARZ  

REAL*8 MNVX1, MNVY1, MNVZ1, SUM, BQBT(50,50)  

REAL*8 TMP1( 50,3), TMP2(3,3), TMP3( 50, 50)  

REAL*8 PK1( 50, 50), G( 50,3)  

REAL*8 DY(3), ES, ED, ESUM, CGN, PRT  

REAL*8 SF, N9, TCHK, ACHK, H1, H2, H3, H4, H5, H6  

REAL*8 AGC(100,100)  

COMPLEX*8 EVAL(100), EVEC (100,100)  

***** VARIABLE DEFINITIONS *****  

STMTRX = SUBROUTINE ESTABLISHES STATE TRANSITION MATRICES  

LAMA = VECTOR OF THE SQUARE OF THE NATURAL FREQUENCIES  

UGVEX = MODE POSITIONS AND SLOPES OF THE NODAL POINTS  

PHI = STATE TRANSITION MATRICES FOR EACH MODE  

GAMMA = INPUT TRANSITION MATRIX  

A = DIAGONAL MATRIX CONSISTING OF PHI  

B = INPUT MATRIX OF GAMMA AND CONTROL SLOPES  

DAMP = DAMPING FACTOR  

SAMPT = SAMPLING TIME

```

C	TCX, TCY, TCZ = CONTROL TORQUE VALUES	GMA00520
C	ENERGY = TOTAL SYSTEM ENERGY	GMA00530
C	IMPLSE = IMPULSE INPUT FUNCTION	GMA00540
C	MIN = NUMBER OF MINUTES SYSTEM WILL BE OBSERVED	GMA00550
C	SMODE = NUMBER OF STARTING MODE (INT)	GMA00560
C	MODE = NUMBER OF MODES (INT)	GMA00570
C	EMODE = NUMBER OF THE LAST MODE (INT)	GMA00580
C	NODE = NUMBER OF THE NOISE INPUT MODE (INT)	GMA00590
C	*** NOISE SLOPE LOCATIONS IN DATA MATRIX ***	GMA00600
C	ROWN1 = X-SLOPE LOCATION	GMA00610
C	ROWN2 = Y-SLOPE LOCATION	GMA00620
C	ROWN3 = Z-SLOPE LOCATION	GMA00630
C	C = OUTPUT MATRIX FOR Y	GMA00640
C	IDENT = IDENTITY MATRIX	GMA00650
C	RMN = MEASUREMENT NOISE COVARIANCE MATRIX	GMA00660
C	QPN = PLANT NOISE COVARIANCE MATRIX	GMA00670
C	PNVARX = PLANT NOISE X-SLOPE VARIANCE	GMA00680
C	PNVARY = PLANT NOISE Y-SLOPE VARIANCE	GMA00690
C	PNVARZ = PLANT NOISE Z-SLOPE VARIANCE	GMA00700
C	MNVARX = MEASUREMENT NOISE X-SLOPE VARIANCE	GMA00710
C	MNVARY = MEASUREMENT NOISE Y-SLOPE VARIANCE	GMA00720
C	MNVARZ = MEASUREMENT NOISE Z-SLOPE VARIANCE	GMA00730
C	ISEED = INITIALIZATION FOR RANDOM NUMBER GENERATOR	GMA00740
C	XKAL = X MATRIX	GMA00750
C	Y = OUTPUT MATRIX	GMA00760
C	RNDM = RANDOM NUMBERS USED FOR WHITE NOISE IN MEASUREMENTS AND IN PLANT FORCES	GMA00770 GMA00780
C	BN = B MATRIX TO MULTIPLY NOISE DISTURBANCES	GMA00790
C	TNX, TNY, TNZ = NOISE TORQUES X, Y, Z SLOPES	GMA00800
C	M2=2*MODE	GMA00810
C		GMA00820
C		GMA00830
C		GMA00840
C	***** SAMPLE OF SPACE EXEC FILE *****	GMA00850 GMA00860
C	THIS FILE MUST BEGIN IN COLUMN 1 AND RUN WITH THE FOLLOWING SEQUENCE FOR THE INITIAL RUN OF THE PROGRAM:	GMA00870 GMA00880
C		GMA00890
C	FORTVS SPACE (COMPILES PROGRAM)	GMA00900
C	SPACE (EXECUTES EXEC FILE)	GMA00910
C	LOAD SPACE (START (LOADS AND EXECUTES PROGRAM)	GMA00920
C		GMA00930
C	SUBSEQUENT PROGRAM RUNS CAN ELIMINATE "FORTVS SPACE" IF NO CHANGES HAVE BEEN MADE TO THE PROGRAM, AND CAN ELIMINATE RUNNING THE EXEC FILE.	GMA00940 GMA00950 GMA00960
C		GMA00970
C	FI 4 DISK THESIS INPUT B (PERM)	GMA00980
C	FI 8 DISK UTILITY DATA (RECFM VS BLOCK 133 PERM	GMA00990
C	FI 11 DISK CNTRL OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM	GMA01000
C	FI 13 DISK GAMMA OUTPUT (RECFM VS BLOCK 133 PERM	GMA01010
C	FI 14 DISK MODE OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM	GMA01020
C	FI 16 DISK COST OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM	GMA01030
C	FI 17 DISK PRT OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM	GMA01040
C	FI 18 DISK ERROR DATA (RECFM F BLOCK 80 LRECL 80 PERM	GMA01050
C	FI 19 DISK END FILE (RECFM F BLOCK 80 LRECL 80 PERM	GMA01060
C	FI 20 DISK GMAT FILE (RECFM F BLOCK 80 LRECL 80 PERM	GMA01070

```

C ***** GMA01080
C ***** GMA01090
C ***** GMA01100
C ***** GMA01110
C ***** GMA01120
C ***** GMA01130
C ***** GMA01140
C ***** GMA01150
C ***** GMA01160
C ***** GMA01170
C ***** GMA01180
C ***** GMA01190
C ***** GMA01200
C ***** GMA01210
C ***** GMA01220
C ***** GMA01230
C ***** GMA01240
C ***** GMA01250
C ***** GMA01260
C ***** GMA01270
C ***** GMA01280
C ***** GMA01290
C ***** GMA01300
C ***** GMA01310
C ***** GMA01320
C ***** GMA01330
C ***** GMA01340
C ***** GMA01350
C ***** GMA01360
C ***** GMA01370
C ***** GMA01380
C ***** GMA01390
C ***** GMA01400
C ***** GMA01410
C ***** GMA01420
C ***** GMA01430
C ***** GMA01440
C ***** GMA01450
C ***** GMA01460
C ***** GMA01470
C ***** GMA01480
C ***** GMA01490
C ***** GMA01500
C ***** GMA01510
C ***** GMA01520
C ***** GMA01530
C ***** GMA01540
C ***** GMA01550
C ***** GMA01560
C ***** GMA01570
C ***** GMA01580
C ***** GMA01590
C ***** GMA01600
C ***** GMA01610
C ***** GMA01620

C
C ***** PARAMETER (JR=5243, KR=5397, MR=262139)
C
C MIN =1200.0
C WT=1.0D00
C PI = 4.0D0 * ATAN(1.0D0)
C
C ***** READ LAMA AND UGVEX MATRICIES *****
C
C
C CALL EXCMS ('CLRSCRN')
C WRITE(6,1008)                                     READING LAMA AND UGVEX MATRICIES'
C WRITE(6,*) '
C WRITE(6,*) '
C THIS SECTION READS THE LAMA VECTOR AND THE UGVEX
C MATRIX AND STORES THEM IN MEMORY FOR FURTHER RECALL OF
C DESIRED LOCATION DATA.
C
C READ(4,1001) NAM
C READ(4,1002)(LAMA(I),I=1,100)
C READ(4,1001) NAM
C DO 5 J = 1,100
C     READ(4,1002)(UGVEX(I,J),I=1,684)
C 5 CONTINUE
C
1001 FORMAT(1X,A6)
1002 FORMAT(1X,8E15.8)
1008 FORMAT(1X,///)
C
500 CALL EXCMS ('CLRSCRN')
C
C ***** STARTING MODE NUMBER ***** GMA01420
C ** SMODE 7 TO 100 (INTEGER) *****
C SMODE=10
C
C WRITE (16,700) SMODE
700 FORMAT (' ','STARTING MODE NUMBER: ',I2)
C
C ***** NUMBER OF MODES TO SCAN ***** GMA01490
C ** MODE 1 TO 93 (INTEGER)
C
C MODE= 3
C
C EMODE = SMODE + MODE - 1
C
C WRITE (16,701) MODE
701 FORMAT (' ','NUMBER OF MODES SCANNED: ',I2)
C
C ***** NOISE INPUT POSITION ***** GMA01590
C ** NODE 1 TO 114 (INTEGER) (IF 0 THEN NO NOISE INPUT)
C NODE= 8
C

```

```

      WRITE (16,702) NODE          GMA01630
702  FORMAT (' ','NOISE NODE LOCATION: ',I5)  CMA01640
C
C
C      *****          SAMPLING TIME          *****
C      ** SAMPT MUST BE LESS THAN OR EQUAL TO SAMPTM **
C      SAMPT = .05          GMA01650
C      SAMPTM = ((2.0D0*PI)/SQRT(LAMA(EMODE)))/2.0D0  GMA01660
C      IF (SAMPT.GE.SAMPTM) THEN  GMA01670
C          SAMPT=SAMPTM  GMA01680
C      ENDIF  GMA01690
C
C      WRITE (16,900) MIN          GMA01700
900  FORMAT (' ',2X,'MIN: ',F8.3)  GMA01710
C
C      WRITE (16,703) SAMPT          GMA01720
703  FORMAT (' ','SAMPLING TIME: ',D12.4)  GMA01730
C
C      *****          DAMPING FACTOR          *****
C      ** DAMP 0.0 TO 1.0 (REAL*8)          GMA01740
C      DAMP=.01  GMA01750
C
C      WRITE (16,704) DAMP          GMA01760
704  FORMAT (' ','DAMPING FACTOR: ',D12.4)  GMA01770
C
C
C      *** PLANT NOISE VARIANCE ***
C      ** PNVARX, PNVARY, PNVARZ GT 0.0          GMA01780
C
C      SF1=2.5D06          GMA01790
C
C      PNVARX=1.0D00*SF1          GMA01800
C      PNVARY=1.0D00*SF1          GMA01810
C      PNVARZ=1.0D00*SF1          GMA01820
C
C
C      *** MEASUREMENT NOISE VARIANCE ***
C      ** MNVARX, MNVARY, MNVARZ GT 0.0          GMA01830
C      SF=1.0          GMA01840
C      MNVX1=5.5D-03*SF          GMA01850
C      MNVY1=5.5D-03*SF          GMA01860
C      MNVZ1=5.5D-03*SF          GMA01870
C
C      510  CALL EXCMS ('CLRSCRN')          GMA01880
C      WRITE (6,1008)          GMA01890
C      WRITE (6,*) '          PROGRAM RUNNING'  GMA01900
C
C      *****          NOISE INPUT LOCATION          *****
C
C      ROWN3 = NODE*6          GMA01910
C      ROWN2 = (NODE*6) - 1          GMA01920
C      ROWN1 = (NODE*6) - 2          GMA01930
C      COUNT = 0          GMA01940
C

```

```

C ***** INITIALIZE MATRICIES ***** GMA02180
C
C DO 40 I = 1,3 GMA02190
  DO 45 J = 1,3 GMA02200
    RMN(I,J)=0.0 GMA02210
45    CONTINUE GMA02220
40    CONTINUE GMA02230
C
C DO 47 I=1,50 GMA02240
  DO 46 J=1,50 GMA02250
    IDENT(I,J)=0.0 GMA02260
    PK(I,J)=0.0 GMA02270
46    CONTINUE GMA02280
47    CONTINUE GMA02290
C
C DO 48 K=1,50 GMA02300
  IDENT(K,K)=1.0 GMA02310
48    CONTINUE GMA02320
C
C *** INITIALIZE RMN AND QPN MATRICES ***
C
C DO 60 I=1,3 GMA02330
  DO 58 J=1,3 GMA02340
    QPN(I,J)=0.0 GMA02350
58    CONTINUE GMA02360
60    CONTINUE GMA02370
C
C RMN(1,1)=MNVX1**2 GMA02380
  RMN(2,2)=MNVY1**2 GMA02390
  RMN(3,3)=MNVZ1**2 GMA02400
  QPN(1,1)=PNVARX**2.0 GMA02410
  QPN(2,2)=PNVARY**2.0 GMA02420
  QPN(3,3)=PNVARZ**2.0 GMA02430
C
C 9999 FORMAT (' ',',',',') GMA02440
C ***** BEGIN MAIN PROGRAM *****
C
C CALL STMTRX(EMODE,SMODE,SAMPT,DAMP,PHI,GAMMA,A,B,LAMA,UGVEX,C, GMA02510
+ ROWN1, ROWN2, ROWN3, BN) GMA02520
C
C ***** PRE-LOOP PORTION OF KALMAN FILTER GMA02530
JK=SMODE*2-2 GMA02540
M2=2*MODE GMA02550
DO 94 I=1,3 GMA02560
  DO 92 J=1,M2
    JL=JK+J
    SUM=0.0
    DO 90 K=1,3
      SUM=SUM+QPN(I,K)*BN(JL,K)
90    CONTINUE
      TMP1(J,I)=SUM
92    CONTINUE
94    CONTINUE

```

```

C GMA02740
C GMA02750
DO 98 I=1,M2 GMA02760
JL=JK+I GMA02770
DO 97 J=1,M2 GMA02780
SUM=0.0 GMA02790
DO 96 K=1,3 GMA02800
SUM=SUM+BN(JL,K)*TMP1(J,K) GMA02810
96 CONTINUE GMA02820
BQBT(I,J)=SUM GMA02830
97 CONTINUE GMA02840
98 CONTINUE GMA02850
C GMA02860
M2=2*MODE GMA02870
DO 100 I=1,M2 GMA02880
DO 99 J=1,M2 GMA02890
TMP3(I,J)=0.0 GMA02900
99 CONTINUE GMA02910
100 CONTINUE GMA02920
JL=JK+M2 GMA02930
DO 9375 I=1,3 GMA02940
DO 9374 J=1,JL GMA02950
C(I,J)=C(I,J)*SF GMA02960
9374 CONTINUE GMA02970
9375 CONTINUE GMA02980
C GMA02990
C **** THIS SECTION COMPUTES THE STATE UPDATE **** GMA03000
C **** THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED **** GMA03010
C **** SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE **** GMA03020
C **** THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED **** GMA03030
ESUM=0.0 GMA03040
COUNT = 0 GMA03050
ENERGY = 0.0DO GMA03060
TIME = 0.0 GMA03070
CGN=0.0 GMA03080
C **** SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE **** GMA03090
C **** THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED **** GMA03100
LOOP = INT((MIN*60.0)/SAMPT) GMA03110
PRT=(DBLE(LOOP))/1200.0 GMA03120
PRTA=(DBLE(LOOP))/2400.0 GMA03130
CNTA=0.0 GMA03140
ACHK=1.0D-10 GMA03150
H1=0.0 GMA03160
H2=0.0 GMA03170
H3=0.0 GMA03180
H4=0.0 GMA03190
H5=0.0 GMA03200
H6=0.0 GMA03210
TCHK=MIN*60.0 GMA03220
9991 CONTINUE GMA03230
C GMA03240
TIME = TIME+ SAMPT GMA03250
C GMA03260
CGN=CGN+1.0 GMA03270
C GNTA=CNIA+1.0 GMA03280

```

```

C **** START OF KALMAN FILTER ****
C
C M2=2*MODE
C
C **** COMPUTATION OF PK*AT ****
C
C JK=2*SMODE-2
DO 175 I=1,M2
    DO 170 J=1,M2
        JL=JK+J
        SUM=0.0
            DO 165 K=1,M2
                JM=JK+K
                SUM =SUM+PK(I,K)*A(JL,JM)
                CONTINUE
                TMP3(I,J)=SUM
                CONTINUE
175 CONTINUE
C
C **** COMPUTATION OF A*(PK*AT)+ BQBT = PK1 ****
C
DO 190 I=1,M2
JL=JK+I
    DO 185 J=1,M2
        SUM=0.0
            DO 180 K=1,M2
                JM=JK+K
                SUM=SUM+A(JL,JM)*TMP3(K,J)
                CONTINUE
                PK1(I,J)=SUM+BQBT(I,J)
                CONTINUE
190 CONTINUE
C **** **** ****
C
C **** COMPUTE PK1*CT ****
C
DO 205 I=1,M2
    DO 200 J=1,3
        SUM=0.0
            DO 195 K=1,M2
                JM=JK+K
                SUM=SUM+PK1(I,K)*C(J,JM)
                CONTINUE
                TMP1(I,J)=SUM
                CONTINUE
200 CONTINUE
C **** **** ****
C
C **** COMPUTE C*(PK1*CT)+RMN ****
DO 220 I=1,3
    DO 215 J=1,3
        SUM=0.0
            DO 210 K=1,M2
                JM=JK+K

```

```

          SUM=SUM+C(I,JM)*TMP1(K,J)          GMA03840
210      CONTINUE
          TMP2(I,J)=SUM+RMN(I,J)          GMA03850
215      CONTINUE
220      CONTINUE
C
C      *** COMPUTATION OF THE INVERSE OF C*PK1*CT + R
C
C      CALL DLINRG ( 3,TMP2,3,TMP2,3)      GMA03860
C
C      *** COMPUTE CT*INV(C*PK1*CT+R)      GMA03870
C
C      DO 245 I=1,M2                    GMA03880
JL=JK+I
      DO 240 J=1,3                    GMA03890
      SUM=0.0
      DO 235 K=1,3                    GMA03900
      SUM=SUM+C(K,JL)*TMP2(K,J)      GMA03910
235      CONTINUE
      TMP1(I,J)=SUM                  GMA03920
240      CONTINUE
245      CONTINUE
C      *****
C
C      *** COMPUTE PK1*C*INV(C*PK1*CT+R) = G *****
C
C      DO 260 I=1,M2                    GMA03930
      DO 255 J=1,3                    GMA03940
      SUM=0.0
      DO 250 K=1,M2                  GMA03950
      SUM=SUM+PK1(I,K)*TMP1(K,J)    GMA03960
250      CONTINUE
      G(I,J)=SUM                  GMA03970
255      CONTINUE
260      CONTINUE
C
      N9=DABS((G(1,1)-H1)/G(1,1))    GMA04000
      IF (N9.GT. ACHK)      THEN      GMA04010
      GO TO 7377
      END IF
      N9=DABS((G(1,3)-H2)/G(1,3))    GMA04020
      IF (N9.GT. ACHK)THEN      GMA04030
      GO TO 7377
      END IF
      N9=DABS((G(2,1)-H3)/G(2,1))    GMA04040
      IF (N9.GT. ACHK)      THEN      GMA04050
      GO TO 7377
      END IF
      N9=DABS((G(2,3)-H4)/G(2,3))    GMA04060
      IF (N9.GT. ACHK)      THEN      GMA04070
      GO TO 7377
      END IF
      N9=DABS((G(3,3)-H5)/G(3,3))    GMA04080
      IF (N9.GT. ACHK)      THEN      GMA04090
      GO TO 7377
      END IF

```

```

        END IF                                GMA04400
        N9=DABS((G(M2,3)-H6)/G(M2,3))          GMA04410
        IF (N9.GT.ACHK)    THEN                 GMA04420
        GO TO 7377                           GMA04430
        END IF                                GMA04440
        GO TO 400                             GMA04450
C
C
7377  CONTINUE                           GMA04460
      H1=G(1,1)                            GMA04470
      H2=G(1,3)                            GMA04480
      H3=G(2,1)                            GMA04490
      H4=G(2,3)                            GMA04500
      H5=G(3,3)                            GMA04510
      H6=G(M2,3)                           GMA04520
                                         GMA04530
                                         GMA04540
                                         GMA04550
                                         GMA04560
                                         GMA04570
IF (TCHK.LE.TIME) THEN                  GMA04580
GO TO 400                                GMA04590
END IF                                    GMA04600
IF (CGN.GE.PRT)  THEN                  GMA04610
C
      WRITE (6,*) 'TIME= ', TIME, ' SEC.'  GMA04620
C
      WRITE (6,*) 'N9= ', N9                GMA04630
      CGN=0.0                               GMA04640
      END IF                                GMA04650
                                         GMA04660
                                         GMA04670
C
C
      *** COMPUTE IDENT - G*C              GMA04680
C
      DO 275 I=1,M2                         GMA04690
        DO 270 J=1,M2                         GMA04700
          JL=JK+J                           GMA04710
          SUM=0.0                            GMA04720
          DO 265 K=1,3                         GMA04730
            SUM=SUM+G(I,K)*C(K,JL)          GMA04740
265      CONTINUE                           GMA04750
          TMP3(I,J)= IDENT(I,J)-SUM        GMA04760
270      CONTINUE                           GMA04770
275      CONTINUE                           GMA04780
C
      *** ****
C
      *** COMPUTE PK= (IDENT - G*C)*PK1  GMA04810
C
      DO 290 I=1,M2                         GMA04820
        DO 285 J=1,M2                         GMA04830
          SUM=0.0                            GMA04840
          DO 280 K=1,M2                         GMA04850
            SUM=SUM+TMP3(I,K)*PK1(K,J)      GMA04860
280      CONTINUE                           GMA04870
          PK(I,J)=SUM                         GMA04880
285      CONTINUE                           GMA04890
290      CONTINUE                           GMA04900
C
      *** ****
C
                                         GMA04910
                                         GMA04920
                                         GMA04930
                                         GMA04940
                                         GMA04950

```

```

C      END OF KALMAN FILTER PART 1 - START OF PART 2 *****
C
C
C      GO TO 9991
C
400  CONTINUE
C
      WRITE (20,1008)
      WRITE (20,*) 'TIME= ',TIME
      DO 384 I=1,M2
      WRITE (20,5350)  G(I,1),G(I,2),G(I,3)
384  CONTINUE
5350 FORMAT (' ',5X,D15.8 ,5X,D15.8 ,5X,D15.8 )
      WRITE (20,*) 'N9= ',N9
C
C      *** COMPUTE AGC = A - G*C
C
      M2=2*MODE
      JK=2*SMODE-2
C
      DO 7155 I=1,M2
      JL=JK+I
      DO 7154 J=1,M2
      JM=JK+J
      SUM=0.0
      DO 7153 K=1,3
      SUM=SUM+G(I,K)*C(K,JM)
7153  CONTINUE
      AGC(I,J)=A(JL,JM)-SUM
7154  CONTINUE
7155  CONTINUE
C
C
C      *** COMPUTE THE EIGENVALUES OF AGC
C
      CALL DEVCRG (M2, AGC, 100, EVAL, EVEC, 100)
C
C      **** PRINT EVAL (EIGENVALUE) MATRIX
C
      DO 7157 I=1,M2
      WRITE (20,*) 'I= ', I, 'EIG= ', EVAL(I)
7157  CONTINUE
C
C
C      STOP
599   END
C
C
C
C
C      **** THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH
C
C
      ****
C
      THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH

```

```

C      OF THE 100 MODES                                GMA05520
C      ****                                              GMA05530
C      ****                                              GMA05540
C      SUBROUTINE STMTRX(EMODE,SMODE,T,D,PHI,GAMMA,A,B,LAMA,UGVEX,C,  GMA05550
C      +      ROWN1,ROWN2,ROWN3,BN)                      GMA05560
C      ****                                              GMA05570
C      REAL*8 WN,GMA,PHI(2,2,100),GAMMA(2,100),EGT,T,COSW1T,SINW1T  GMA05580
C      REAL*8 W1,D,A(200,200),B(200,3),C(6,200),BN(200,3)          GMA05590
C      REAL LAMA(100),UGVEX(684,100)                      GMA05600
C      INTEGER SMODE,R,EMODE,JJ,KK,ROWN1,ROWN2,ROWN3            GMA05610
C      ****                                              GMA05620
C      ****                                              GMA05630
C      ****                                              GMA05640
C      ****                                              GMA05650
C      ****                                              GMA05660
C      DO 600 I = 1      ,100                           GMA05670
C      WN = DBLE(SQRT(LAMA(I)))                      GMA05680
C      GMA = D*WN/2.0                                GMA05690
C      EGT = DEXP(-GMA*T)                           GMA05700
C      W1 = DSQRT((WN**2)-(GMA**2))                GMA05710
C      COSW1T = DCOS(W1*T)                           GMA05720
C      SINW1T = DSIN(W1*T)                           GMA05730
C      ****                                              GMA05740
C      ****                                              GMA05750
C      ****                                              GMA05760
C      ****                                              GMA05770
C      ****                                              GMA05780
C      IF(WN.EQ.0)THEN                                GMA05790
C          PHI(1,1,I) = EGT*COSW1T                  GMA05800
C          PHI(1,2,I) = T                           GMA05810
C          PHI(2,1,I) = 0                           GMA05820
C          PHI(2,2,I) = EGT*COSW1T                  GMA05830
C      ****                                              GMA05840
C      ****                                              GMA05850
C      ****                                              GMA05860
C      ****                                              GMA05870
C      ****                                              GMA05880
C      ****                                              GMA05890
C      GAMMA(1,I) = 0                                GMA05900
C      GAMMA(2,I) = 0                                GMA05910
C      ELSE                                              GMA05920
C      ****                                              GMA05930
C      ****                                              GMA05940
C      ****                                              GMA05950
C      ****                                              GMA05960
C      ****                                              GMA05970
C      PHI(1,1,I) = EGT*(COSW1T + (GMA*(W1**(-1)))*SINW1T)  GMA05980
C      PHI(1,2,I) = (W1**(-1))*EGT*SINW1T            GMA05990
C      PHI(2,1,I) = -(WN**2)*(W1**(-1))*EGT*SINW1T  GMA06000
C      PHI(2,2,I) = EGT*(COSW1T - (GMA*(W1**(-1)))*SINW1T)  GMA06010
C      ****                                              GMA06020
C      ****                                              GMA06030
C      ****                                              GMA06040
C      ****                                              GMA06050
C      GAMMA(1,I)=(WN**(-2))*(1.0-EGT*COSW1T-EGT*(GMA/W1)*SINW1T)  GMA06060

```

```

C GAMMA(2,I) = (W1**(-1))*EGT*SINW1T
C
C
C
C      ENDIF
C
C
C
600  CONTINUE
C
C
C
C
C      R = 1
C
DO 610 K = 1      ,100
C
C
C
C
C
A(R,R) = PHI(1,1,K)
A(R,R+1) = PHI(1,2,K)
A(R+1,R) = PHI(2,1,K)
A(R+1,R+1) = PHI(2,2,K)
C
C
C
C
C
C
C      *** B MATRIX FOR MULTIPLYING CONTROL TORQUES
C
B(R,1) = GAMMA(1,K)*DBLE(UGVEX(412,K))
B(R,2) = GAMMA(1,K)*DBLE(UGVEX(413,K))
B(R,3) = GAMMA(1,K)*DBLE(UGVEX(414,K))
B(R+1,1) = GAMMA(2,K)*DBLE(UGVEX(412,K))
B(R+1,2) = GAMMA(2,K)*DBLE(UGVEX(413,K))
B(R+1,3) = GAMMA(2,K)*DBLE(UGVEX(414,K))
C
C
C
C
C
C
C
C      *** BN MATRIX FOR MULTIPLYING THE NOISE DISTURBANCES
C
BN(R,1)=GAMMA(1,K)*DBLE(UGVEX(ROWN1,K))
BN(R,2)=GAMMA(1,K)*DBLE(UGVEX(ROWN2,K))
BN(R,3)=GAMMA(1,K)*DBLE(UGVEX(ROWN3,K))
BN(R+1,1)=GAMMA(2,K)*DBLE(UGVEX(ROWN1,K))
BN(R+1,2)=GAMMA(2,K)*DBLE(UGVEX(ROWN2,K))

```

```

C BN(R+1,3)=GAMMA(2,K)*DBLE(UGVEX(ROWN3,K)) GMA06630
C GMA06640
C GMA06650
C GMA06660
C GMA06670
C GMA06680
C GMA06690
C GMA06700
C R = R+2 GMA06710
610  CONTINUE GMA06720
C GMA06730
C GMA06740
C GMA06750
C GMA06760
C GMA06770
C GMA06780
C *** C MATRIX PRODUCTION ***
C GMA06790
C GMA06800
C GMA06810
C GMA06820
C JJ=-1 GMA06830
DO 640 I=1,100 GMA06840
JJ=JJ+1 GMA06850
KK=I+JJ GMA06860
C GMA06870
C GMA06880
C C(1,KK) = DBLE(UGVEX(418,I)) GMA06890
C C(2,KK) = DBLE(UGVEX(419,I)) GMA06900
C C(3,KK) = DBLE(UGVEX(420,I)) GMA06910
C GMA06920
C GMA06930
C GMA06940
C KK=KK+1 GMA06950
C GMA06960
C C(1,KK)=0.0 GMA06970
C C(2,KK)=0.0 GMA06980
C C(3,KK)=0.0 GMA06990
C 640  CONTINUE GMA07000
C GMA07010
C GMA07020
C GMA07030
C GMA07040
C RETURN GMA07050
C END GMA07060

```

## APPENDIX B. KALMAN OBSERVER AND PLANT SIMULATION

```

C ***** SIMRUN ***** SIM00010
C ***** SIM00020
C ***** SIM00030
C ***** ADAPTED TO READ KALMAN FILETER G MATRICE ***** SIM00040
C ***** THEN RUN ALL N MODES OF THE PLANT WHILE ***** SIM00050
C ***** USING A KALMAN FILTER TO OBSERVE M ***** SIM00060
C ***** NUMBER OF STATES ***** SIM00070
C ***** SIM00080
C ***** SIM00090
C ***** SIM00100
C ***** SIM00110
C ***** VARIABLE DECLARATIONS ***** SIM00120
C ***** SIM00130
C ***** SIM00140
C EXTERNAL STMTRX,EXCMS SIM00150
C CHARACTER*6 NAM SIM00160
C CHARACTER*1 AGAIN, CORECT, RAGAIN SIM00170
C INTEGER ROWN1, ROWN2, ROWN3, COUNT, NODE, MODE, KQ, EMODE, SMODE, R2M, C2M SIM00180
C INTEGER CT, CF, KADJ, CFADJ, LOOP, PRNT, JJ, JK, N1, JR, KR, MR, ISEED, M2 SIM00190
C INTEGER ITYPE(200), IPVT(100), NS, NF, SN, FN SIM00200
C INTEGER JL, J1, JM, JP, JQ, KA, KB, KC, KD, KE, KF, KG SIM00210
C ***** SIM00220
C ***** SIM00230
C ***** SIM00240
C REAL LAMA(100), UGVEX(684,100), RNODE, RMODE, MIN SIM00250
C REAL*8 PHI(2,2,100), GAMMA(2,100), EGT, GMA, WN, W1, X1T, X2T, TIME SIM00260
C REAL*8 A(200,200), B(200,3), F(3, 50), IMPLSE, ENERGY SIM00270
C REAL*8 COSW1T, SINW1T, X(200) SIM00280
C REAL*8 TCX, TCY, TCZ, DAMP, SAMPT, PI, SAMPTM, SUM1, SUM2, SUM3, SUMC SIM00290
C REAL*8 C(3,200), IDENT( 50, 50), RMN(3,3), QPN(3,3) SIM00300
C REAL*8 Y(3) , BN(200,3) SIM00310
C REAL*8 PNVARX, PNVARY, PNVARZ SIM00320
C REAL*8 MNVARX, MNVARY, MNVARZ SIM00330
C REAL*8 SUM, RNDM(6), RND1, RND2 SIM00340
C REAL*8 XH( 50) , BQBT( 50, 50) SIM00350
C REAL*8 SF1 SIM00360
C REAL*8 TMP1( 50,3), TMP2(3,3), TMP3( 50, 50) SIM00370
C REAL*8 G( 50,3) SIM00380
C REAL*8 XH1( 50) , DY(3) , ES, ED, ESUM, CGN, PRT SIM00390
C REAL*8 WT , WD(3), BNWD(200) SIM00400
C REAL*8 AX(200) , V(3), SF , TO, CTT, ESS SIM00410
C REAL*8 CTG, XDEL, E2(100), XDEL1, ERS, PRT1, E3(100), XS(100) SIM00420
C ***** SIM00430
C ***** SIM00440
C ***** VARIABLE DEFINITIONS ***** SIM00450
C ***** SIM00460
C ***** SIM00470
C STMTRX = SUBROUTINE EXTABLISHES STATE TRANSITION MATRICIES SIM00480
C LAMA = VECTOR OF THE SQUARE OF THE NATURAL FREQUENCIES SIM00490
C UGVEX = MODE POSITONS AND SLOPES OF THE NODAL POINTS SIM00500
C PHI = STATE TRANSITION MATRICIES FOR EACH MODE SIM00510

```

C GAMMA = INPUT TRANSITION MATRIX	SIM00520
C A = DIAGONAL MATRIX CONSISTING OF PHI	SIM00530
C B = INPUT MATRIX OF GAMMA AND CONTROL SLOPES	SIM00540
C DAMP = DAMPING FACTOR	SIM00550
C SAMPT = SAMPLING TIME	SIM00560
C TCX, TCY, TCZ = CONTROL TORQUE VALUES	SIM00570
C ENERGY = TOTAL SYSTEM ENERGY	SIM00580
C IMPLSE = IMPULSE INPUT FUNCTION	SIM00590
C MIN = NUMBER OF MINUTES SYSTEM WILL BE OBSERVED	SIM00600
C SMODE = NUMBER OF STARTING MODE (INT)	SIM00610
C MODE = NUMBER OF MODES (INT)	SIM00620
C EMODE = NUMBER OF THE LAST MODE (INT)	SIM00630
C NODE = NUMBER OF THE NOISE INPUT MODE (INT)	SIM00640
C *** NOISE SLOPE LOCATIONS IN DATA MATRIX ***	SIM00650
C ROWN1 = X-SLOPE LOCATION	SIM00660
C ROWN2 = Y-SLOPE LOCATION	SIM00670
C ROWN3 = Z-SLOPE LOCATION	SIM00680
C C = OUTPUT MATRIX FOR Y	SIM00690
C IDENT = IDENTITY MATRIX	SIM00700
C RMN = MEASUREMENT NOISE COVARIANCE MATRIX	SIM00710
C QPN = PLANT NOISE COVARIANCE MATRIX	SIM00720
C PNVARX = PLANT NOISE X-SLOPE VARIANCE	SIM00730
C PNVARY = PLANT NOISE Y-SLOPE VARIANCE	SIM00740
C PNVARZ = PLANT NOISE Z-SLOPE VARIANCE	SIM00750
C MNVARX = MEASUREMENT NOISE X-SLOPE VARIANCE	SIM00760
C MNVARY = MEASUREMENT NOISE Y-SLOPE VARIANCE	SIM00770
C MNVARZ = MEASUREMENT NOISE Z-SLOPE VARIANCE	SIM00780
C ISEED = INITIALIZATION FOR RANDOM NUMBER GENERATOR	SIM00790
C XKAL = X MATRIX	SIM00800
C Y = OUTPUT MATRIX	SIM00810
C RNDM = RANDOM NUMBERS USED FOR WHITE NOISE IN MEASUREMENTS AND	SIM00820
C IN PLANT FORCES	SIM00830
C BN = B MATRIX TO MULTIPLY NOISE DISTURBANCES	SIM00840
C TNX, TNY, TNZ = NOISE TORQUES X, Y, Z SLOPES	SIM00850
C M2=2*MODE	SIM00860
C	SIM00870
C	SIM00880
C ***** SAMPLE OF SPACE EXEC FILE *****	SIM00900
C	SIM00910
C THIS FILE MUST BEGIN IN COLUMN 1 AND RUN WITH THE FOLLOWING	SIM00920
C SEQUENCE FOR THE INITIAL RUN OF THE PROGRAM:	SIM00930
C	SIM00940
C FORTVS SPACE (COMPILES PROGRAM)	SIM00950
C SPACE (EXECUTES EXEC FILE)	SIM00960
C LOAD SPACE (START (LOADS AND EXECUTES PROGRAM)	SIM00970
C	SIM00980
C SUBSEQUENT PROGRAM RUNS CAN ELIMINATE "FORTVS SPACE" IF NO	SIM00990
C CHANGES HAVE BEEN MADE TO THE PROGRAM, AND CAN ELIMINATE	SIM01000
C RUNNING THE EXEC FILE.	SIM01010
C	SIM01020
C FI 4 DISK THESIS INPUT B (PERM)	SIM01030
C FI 8 DISK UTILITY DATA (RECFM VS BLOCK 133 PERM)	SIM01040
C FI 11 DISK CNTRL OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM)	SIM01050
C FI 13 DISK GAMMA OUTPUT (RECFM VS BLOCK 133 PERM)	SIM01060
C FI 14 DISK MODE OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM)	SIM01070

```

C   FI 16 DISK COST OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM      SIM01080
C   FI 17 DISK PRT OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM      SIM01090
C   FI 18 DISK ERROR DATA  (RECFM F BLOCK 80 LRECL 80 PERM      SIM01100
C   FI 19 DISK END FILE   (RECFM F BLOCK 80 LRECL 80 PERM      SIM01110
C   FI 20 DISK GMAT FILE  (RECFM F BLOCK 80 LRECL 80 PERM      SIM01120
C
C   ****
C
C   PARAMETER (JR=5243, KR=5397, MR=262139)                      SIM01130
C
C
C   MIN =1.00
C
C   WT=1.0D00
C   PI = 4.0D0 * ATAN(1.0D0)
C
C   ****
C   **** READ LAMA AND UGVEX MATRICIES ****
C   ****
C
C   CALL EXCMS ('CLRSCRN')
C   WRITE(6,1008)
C   WRITE(6,*) '          READING LAMA AND UGVEX MATRICIES'
C   WRITE(6,*) ' '
C
C   THIS SECTION READS THE LAMA VECTOR AND THE UGVEX
C   MATRIX AND STORES THEM IN MEMORY FOR FURTHER RECALL OF
C   DESIRED LOCATION DATA.
C
C   READ(4,1001) NAM
C   READ(4,1002)(LAMA(I),I=1,100)
C   READ(4,1001) NAM
C   DO 5 J = 1,100
C       READ(4,1002)(UGVEX(I,J),I=1,684)
C
5   CONTINUE
C
1001 FORMAT(1X,A6)
1002 FORMAT(1X,8E15.8)
1008 FORMAT (1X,///)
C
500  CALL EXCMS ('CLRSCRN')
C
C   **** STARTING MODE NUMBER ****
C   ** SMODE 7 TO 100 (INTEGER) ****
C   SMODE= 7
C
C   WRITE (16,700) SMODE
700  FORMAT (' ', 'STARTING MODE NUMBER: ',I2)
C
C   **** NUMBER OF MODES TO SCAN ****
C   ** MODE 1 TO 93 (INTEGER)
C
C   MODE=20
C
C   EMODE = SMODE + MODE - 1
C

```

```

    WRITE (16,701) MODE           SIM01630
701  FORMAT (' ','NUMBER OF MODES SCANNED: ',I12)      SIM01640
C
C  ***** NOISE INPUT POSITION      *****
C  ** NODE 1 TO 114 (INTEGER) (IF 0 THEN NO NOISE INPUT)  SIM01650
C  NODE= 8                                         SIM01660
C
C  WRITE (16,702) NODE           SIM01670
702  FORMAT (' ','NOISE NODE LOCATION: ',I15)      SIM01680
C
C  ***** START AND STOP FOR PLANT  SIM01690
SN=7
FN=20
NS=SN*2-1
NF=SN*2+2*FN-2
WRITE (16,899) SN,FN           SIM01700
899  FORMAT (' ','PLANT -- SN= ',I5,' FN= ',I5)      SIM01710
C  ***** SAMPLING TIME          *****
C  ** SAMPT MUST BE LESS THAN OR EQUAL TO SAMPTM **  SIM01720
C  SAMPT = 0.05                                         SIM01730
SAMPTM = ((2.0D0*PI)/SQRT(LAMA(EMODE)))/1.0D01      SIM01740
IF (SAMPT.GE.SAMPTM) THEN
  SAMPT=SAMPTM
ENDIF
C
C  WRITE (16,900) MIN           SIM01750
900  FORMAT (' ',2X,'MIN: ',F8.3)      SIM01760
C
C  WRITE (16,703) SAMPT, SAMPTM      SIM01770
703  FORMAT (' ','SAMPLING TIME: ',D12.4,2X,'SAMPTM= ',D15.8)  SIM01780
C
C  ***** DAMPING FACTOR          *****
C  ** DAMP 0.0 TO 1.0 (REAL*8)    SIM01790
C  DAMP=.01                                         SIM01800
C
C  WRITE (16,704) DAMP           SIM01810
704  FORMAT (' ','DAMPING FACTOR: ',D12.4)      SIM01820
C
C
C  *** PLANT NOISE VARIANCE ***
C  ** PNVARX, PNVARY, PNVARZ GT 0.0  SIM01830
SF1=2.5D06
SF=1.0D00
C
C  PNVARX=1.0D00*SF1
PNVARY=1.0D00*SF1
PNVARZ=1.0D00*SF1
C
C
C  *** MEASUREMENT NOISE VARIANCE ***
C  ** MNVARX, MNVARY, MNVARZ GT 0.0  SIM01840
MNVARX=1.0D-03 *SF
MNVARY=1.0D-03 *SF

```

```

MNVARZ=1.0D-03 *SF SIM02170
C SIM02180
C SIM02190
C SIM02200
711 WRITE (16,711) SIM02210
    FORMAT(' ', 'PLANT NOISE VARIANCE: ')
    WRITE (16,712) SIM02220
    FORMAT(' ',6X,'PNVARX',13X,'PNVARY',13X,'PNVARZ') SIM02230
    WRITE (16,713) PNVARX, PNVARY, PNVARZ SIM02240
    FORMAT(' ',2X,E15.8,2X,E15.8,2X,E15.8) SIM02250
    WRITE(16,714) SIM02260
    FORMAT(' ', 'MEASUREMENT NOISE: ')
    WRITE(16,715) SIM02270
    FORMAT(' ',6X,'MNVARX',13X,'MNVARY',13X,'MNVARZ') SIM02280
    WRITE(16,713) MNVARX,MNVARY,MNVARZ SIM02290
C SIM02300
510 CALL EXCMS ('CLRSCRN') SIM02310
    WRITE (6,1008) SIM02320
    WRITE (6,*) ' SIM02330
C PROGRAM RUNNING' SIM02340
C ***** NOISE INPUT LOCATION ***** SIM02350
C ***** SIM02360
C ***** SIM02370
C ROWN3 = NODE*6 SIM02380
C ROWN2 = (NODE*6) - 1 SIM02390
C ROWN1 = (NODE*6) - 2 SIM02400
C COUNT = 0 SIM02410
C SIM02420
C SIM02430
C ***** INITIALIZE MATRICIES ***** SIM02440
C ***** SIM02450
C DO 48 K=1,50 SIM02460
    IDENT(K,K)=1.0 SIM02470
48 CONTINUE SIM02480
C SIM02490
C DO 54 K = 1, 200 SIM02500
    X(K) = 0.0 SIM02510
54 CONTINUE SIM02520
C SIM02530
C SIM02540
C WRITE(6,1008) SIM02550
C WRITE (6,*) ' INITIALIZE RMN AND QPN MATRICES ' SIM02560
C *** INITIALIZE RMN AND QPN MATRICES *** SIM02570
C SIM02580
C DO 60 I=1,3 SIM02590
    DO 58 J=1,3 SIM02600
        RMN(I,J)=0.0 SIM02610
        QPN(I,J)=0.0 SIM02620
58 CONTINUE SIM02630
60 CONTINUE SIM02640
C SIM02650
C RMN(1,1)=MNVARX**2 SIM02660
C RMN(2,2)=MNVARY**2 SIM02670
C RMN(3,3)=MNVARZ**2 SIM02680
C QPN(1,1)=PNVARX**2 SIM02690
C QPN(2,2)=PNVARY**2 SIM02700

```

```

C QPN(3,3)=PNVARZ**2.0 SIM02710
C
C WRITE(6,1008) SIM02720
C WRITE(6,*) ' ENTER STMTRX SIM02730
C ***** BEGIN MAIN PROGRAM ***** SIM02740
C ***** SIM02750
C ***** SIM02760
C ***** SIM02770
C ***** SIM02780
C ***** SIM02790
C ***** SIM02800
C CALL STMTRX(EMODE,SMODE,SAMPT,DAMP,PHI,GAMMA,A,B,LAMA,UGVEX,C, SIM02810
+ ROWN1,ROWN2,ROWN3,BN) SIM02820
C
C WRITE (16,1008) SIM02830
DO 11000 I=1,200 SIM02840
DO 10900 J=1,3 SIM02850
C(J,I)= C( J,I)*SF SIM02860
10900 CONTINUE SIM02870
11000 CONTINUE SIM02880
C
C *** PRE-LOOP PORTION OF KALMAN FILTER SIM02890
C
M2=2*MODE SIM02900
JP=2*SMODE-1 SIM02910
JQ=2*EMODE SIM02920
DO 90 I=1,50 SIM02930
XH(I)=0.0 SIM02940
90 CONTINUE SIM02950
C
DO 9971 I=1,M2 SIM02960
READ (20,*) G(I,1), G(I,2), G(I,3) SIM02970
9971 CONTINUE SIM02980
C
C
WRITE (14,1008) SIM02990
DO 384 I=1,M2 SIM03000
WRITE (14,5350) G(I,1),G(I,2),G(I,3) SIM03010
384 CONTINUE SIM03020
5350 FORMAT (' ',2X,D15.8,2X,D15.8,2X,D15.8) SIM03030
C
C
***** THIS SECTION COMPUTES THE STATE UPDATE ***** SIM03040
C ***** SIM03050
DO 9771 I=1,100 SIM03060
E2(I)=0.0 SIM03070
E3(I)=0.0 SIM03080
XS(I)=0.0 SIM03090
9771 CONTINUE SIM03100
ESS =0.0 SIM03110
COUNT = 0 SIM03120
ENERGY = 0.000 SIM03130
TIME = 0.0 SIM03140
CGN=0.0 SIM03150
SIM03160
SIM03170
SIM03180
SIM03190
SIM03200
SIM03210
SIM03220
SIM03230
SIM03240
SIM03250

```

```

CTG=0.0                                SIM03260
C ***** SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE *****  SIM03270
C ***** THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED *****  SIM03280
C WRITE (6,1008)                         SIM03290
C WRITE (6,*) '                      START STATE UPDATE
C LOOP = INT((MIN*60.0)/SAMPT)          SIM03310
C PRT= (DBLE(LOOP))/30.0                SIM03320
C CTT=0.0                                SIM03330
C DO 400 L = 0, LOOP                   SIM03340
C     TIME = DBLE(L)*SAMPT              SIM03350
C
C     IF(L.EQ.0)THEN                   SIM03360
C         IMPLSE =(1.0D06*SF1)/(DSQRT(SAMPT))  SIM03370
C     ELSE                           SIM03380
C         IMPLSE = 0.0D0              SIM03390
C     ENDIF                         SIM03400
C
C     TO=0.0                           SIM03410
C ***** RANDOM NUMBER GENERATOR *****  SIM03420
C
C DO 101 I=1,6                         SIM03430
C     ISEED=MOD(ISEED*JR+KR,MR)        SIM03440
C     RND1=(DBLE(ISEED)+0.5D00)/DBLE(MR)  SIM03450
C     ISEED=MOD(ISEED*JR+KR,MR)        SIM03460
C     RND2=(DBLE(ISEED)+0.5D00)/DBLE(MR)  SIM03470
C     RNDM(I)=DSQRT(-2.0*DLOG(RND1))*DCOS(6.2831853D00*RND2)  SIM03480
101  CONTINUE                         SIM03490
C ***** ***** ***** ***** ***** *****  SIM03500
C     CTT=CTT+1.0                     SIM03510
C     ***** START OF STATE UPDATE ****  SIM03520
C
C     *** COMPUTE AX^200 = A^200 X 200 * X^200  SIM03530
C
C     *** COMPUTE AX = A*X              SIM03540
C
C     JK=SMODE*2-2                   SIM03550
C     JP=JK+1                        SIM03560
C     JQ=2*EMODE                      SIM03570
C
C     DO 5015 I=NS,NF                SIM03580
C         SUM=0.0                      SIM03590
C             DO 5010 K=NS,NF          SIM03600
C                 SUM=SUM+A(I,K)*X(K)  SIM03610
5010  CONTINUE                         SIM03620
C                 AX(I)=SUM          SIM03630
5015  CONTINUE                         SIM03640
C
C     *** COMPUTE WD^3                SIM03650
C
C     WD(1)=PNVARX*RNDM(1)*TO+IMPLSE  SIM03660
C     WD(2)=PNVARY*RNDM(2)*TO          SIM03670
C     WD(3)=PNVARZ*RNDM(3)*TO          SIM03680
C
C
C

```

```

C          **** COMPUTE BNWD0200 =BN0200 X 3 * WD03          SIM03820
C
C          DO 5035 I=NS,NF          SIM03830
C          SUM=0.0          SIM03840
C          DO 5030 K=1,3          SIM03850
C          SUM=SUM+BN(I,K)*WD(K)          SIM03860
C          CONTINUE          SIM03870
C          BNWD(I)=SUM          SIM03880
5030      CONTINUE          SIM03890
C          SIM03900
C          **** COMPUTE X0200 =AX0200 + BNWD0200          SIM03910
C
C          DO 5040 I=NS,NF          SIM03920
C          X(I)= AX(I) + BNWD(I)          SIM03930
C          IF (DABS(X(I)).LT. 1.0D-60) THEN          SIM03940
C          X(I)=1.0D-60          SIM03950
C          END IF          SIM03960
C
C          CONTINUE          SIM03970
C          SIM03980
C          SIM03990
C          SIM04000
C          SIM04010
C          SIM04020
5040      CONTINUE          SIM04030
C          **** COMPUTE V03          SIM04040
C
C          V(1)=MNVARX*RNDM(4)          SIM04050
C          V(2)=MNVARY*RNDM(5)          SIM04060
C          V(3)=MNVARZ*RNDM(6)          SIM04070
C
C          **** COMPUTE Y03 = C03 X 200 * X0200 + V03          SIM04080
C
C          DO 5050 I=1,3          SIM04090
C          SUM=0.0          SIM04100
C          DO 5045 K=NS,NF          SIM04110
C          SUM=SUM+C(I,K)*X(K)          SIM04120
C          CONTINUE          SIM04130
C          Y(I)=SUM+V(I)          SIM04140
5045      CONTINUE          SIM04150
C          SIM04160
C          SIM04170
C          SIM04180
C          SIM04190
C          SIM04200
C          SIM04210
C          SIM04220
C          **** START OF KALMAN FILTER ****          SIM04230
C
C          M2=2*MODE          SIM04240
C
C          **** COMPUTE XH1 = A*XH          SIM04250
C
C          DO 300 I=JP,JQ          SIM04260
C          SUM=0.0          SIM04270
C          DO 295 K=JP,JQ          SIM04280
C          SUM=SUM+A(I,K) * XH(K)          SIM04290
295      CONTINUE          SIM04300
C          XH1(I)=SUM          SIM04310
C          CONTINUE          SIM04320
C          SIM04330
C          SIM04340
C          SIM04350
C          SIM04360
C          ****          SIM04370

```

```

C      **** COMPUTE DY = Y - C*XH1          SIM04380
C
C      DO 315 I=1,3                         SIM04390
C      SUM=0.0                               SIM04400
C          DO 310 K=JP,JQ                   SIM04410
C          SUM=SUM+C(I,K)*XH1(K)           SIM04420
310      CONTINUE                           SIM04430
C          DY(I)=Y(I)-SUM                 SIM04440
315      CONTINUE                           SIM04450
C
C      **** **** ****
C
C      **** COMPUTE XH = XH1 + G*DY          SIM04460
C
C      DO 325 I=1,M2                         SIM04470
C          J1=JP-1+I                         SIM04480
C          SUM=0.0                           SIM04490
C              DO 320 K=1,3                 SIM04500
C              SUM=SUM+G(I,K)*DY(K)         SIM04510
320      CONTINUE                           SIM04520
C          XH(J1)=XH1(J1)+SUM             SIM04530
C          IF (DABS(XH(J1)).LT.1.0D-60) THEN
C              XH(J1)=1.0*D-60           SIM04540
C          END IF                           SIM04550
C
325      CONTINUE                           SIM04560
C
C
C      **** END OF KALMAN ROUTINES ****
C
C      *** COMPUTATION OF ESUM ***
C
C      DO 340 I=JP,JQ                      SIM04630
C          XDEL= X(I)-XH(I)                SIM04640
C          XDEL1=XDEL*XDEL*SAMPT          SIM04650
C          E2(I)=E2(I)+XDEL1             SIM04660
C          XS(I)=XS(I)+X(I)*X(I)*SAMPT  SIM04670
C          E3(I)=E2(I)/XS(I)             SIM04680
340      CONTINUE                           SIM04690
C
C          CGN=CGN+1.0                     SIM04700
C          IF (CTT.EQ.1.0.OR.CGN.GT.PRT) THEN
C
C              WRITE (6,*) 'TIME= ', TIME, ' SEC.'
C
C              WRITE (17,1008)                 SIM04710
C              WRITE (16,1008)                 SIM04720
C              WRITE (16,2100) TIME           SIM04730
C
C              WRITE (17,2100) TIME           SIM04740
2100      FORMAT(' ', 'TIME= ', F9.3)      SIM04750
C          DO 380 I=JP, JQ                 SIM04760
C              WRITE (16,4500) I,X(I) ,I ,XH(I)
C
C

```

```

      WRITE (17,4530) I,E2(I) ,E3(I) , XS(I)           SIM04930
380  CONTINUE                                         SIM04940
C                                         SIM04950
C                                         SIM04960
C                                         SIM04970
C                                         SIM04980
CGN=0.0                                         SIM04990
END IF                                         SIM05000
4500 FORMAT (' ','X('',I3,'')= ',D15.8,2X,'XH('',I3,'')= ',D15.8) SIM05010
4530 FORMAT (' ',5X,I5,5X,3 D15.8)                 SIM05020
C                                         SIM05030
400  CONTINUE                                         SIM05040
C                                         SIM05050
C                                         SIM05060
DO 401 I=JP,JQ                                 SIM05070
      WRITE (19,4530) I, E2(I) ,E3(I), XS(I)          SIM05080
401  CONTINUE                                         SIM05090
C                                         SIM05100
C                                         SIM05110
C                                         SIM05120
C                                         SIM05130
599  STOP                                         SIM05140
      END                                         SIM05150
C                                         SIM05160
C                                         SIM05170
C                                         SIM05180
THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH SIM05190
OF THE 100 MODES                                         SIM05200
SUBROUTINE STMTRX(EMODE,SMODE,T,D,PHI,GAMMA,A,B,LAMA,UGVEX,C, SIM05210
+      ROWN1,ROWN2,ROWN3,BN)                           SIM05220
REAL*8 WN,GMA,PHI(2,2,100),GAMMA(2,100),EGT,T,COSW1T,SINW1T SIM05230
REAL*8 W1,D,A(200,200),B(200,3),C(3,200),BN(200,3)          SIM05240
REAL LAMA(100),UGVEX(684,100)                         SIM05250
INTEGER SMODE,R,EMODE,JJ,KK,ROWN1,ROWN2,ROWN3             SIM05260
DO 600 I = 1 ,100                                     SIM05270
      WN = DBLE(SQRT(LAMA(I)))                      SIM05280
      GMA = D*WN/2.0                                 SIM05290
      EGT = DEXP(-GMA*T)                           SIM05300
      W1 = DSQRT((WN**2)-(GMA**2))                 SIM05310
      COSW1T = DCOS(W1*T)                           SIM05320
      SINW1T = DSIN(W1*T)                           SIM05330
IF(WN.EQ.0)THEN                                     SIM05340
      PHI(1,1,I) = EGT*COSW1T                      SIM05350
      PHI(1,2,I) = T                               SIM05360
      PHI(2,1,I) = 0                               SIM05370
      PHI(2,2,I) = EGT*COSW1T                      SIM05380
IF(WN.EQ.0)THEN                                     SIM05390
      PHI(1,1,I) = EGT*COSW1T                      SIM05400
      PHI(1,2,I) = T                               SIM05410
      PHI(2,1,I) = 0                               SIM05420
      PHI(2,2,I) = EGT*COSW1T                      SIM05430
      PHI(1,1,I) = EGT*COSW1T                      SIM05440
      PHI(1,2,I) = T                               SIM05450
      PHI(2,1,I) = 0                               SIM05460
      PHI(2,2,I) = EGT*COSW1T                      SIM05470
      PHI(1,1,I) = EGT*COSW1T                      SIM05480

```





C  
C

RETURN  
END

SIM06610  
SIM06620  
SIM06630  
SIM06640

## APPENDIX C. PROGRAM TO ESTIMATE NOISE IN KALMAN FILTER FROM UNOBSERVED MODES

```

***** SPAC 24 *****
***** ADAPTED TO RUN N MODES OF THE PLANT AND *****
***** COMPUTE THE NOISE IN THE KALMAN FILTER *****
***** FROM THE UNOBSERVED MODES *****
***** VARIABLE DECLARATIONS *****
EXTERNAL STMTRX,EXCMS
CHARACTER*6 NAM
CHARACTER*1 AGAIN, CORECT, RAGAIN
INTEGER ROWN1, ROWN2, ROWN3, COUNT, NODE, MODE, KQ, EMODE, SMODE, R2M, C2M
INTEGER CT, CF, KADJ, CFADJ, LOOP, PRNT, JJ, JK, N1, JR, KR, MR, ISEED, M2
INTEGER NO, NS, NF, SN, FN
INTEGER JL, J1, JM, JP, JQ, KA, KB, KC, KD, KE, KF, KG
REAL LAMA(100), UGVEX(684,100), RNODE, RMODE, MIN
REAL*8 PHI(2,2,100), GAMMA(2,100), EGT, GMA, WN, W1, X1T, X2T, TIME
REAL*8 A(200,200), B(200,3), F(3, 50), IMPLSE, ENERGY
REAL*8 COSW1T, SINW1T, X(200)
REAL*8 DAMP, SAMPT, PI, SAMPTM, SUM1, SUM2, SUM3, SUMC
REAL*8 C(9,200), RMN(3,3), QPN(3,3)
REAL*8 BN(200,3)
REAL*8 PNVARX, PNVARY, PNVARZ
REAL*8 MNVARX, MNVARY, MNVARZ
REAL*8 SUM, RNDM(6), RND1, RND2
REAL*8 ES, ED, ESUM, CGN, PRT
REAL*8 WT, WD(3), BNWD(200), EX1(9)
REAL*8 EX(9), AX(200), SF, TO, CTT, ESS
REAL*8 CTG, XDEL, XDEL1, ERS, PRT1
REAL*8 SF1
***** VARIABLE DEFINITIONS *****
STMTRX = SUBROUTINE ESTABLISHES STATE TRANSITION MATRICES
LAMA = VECTOR OF THE SQUARE OF THE NATURAL FREQUENCIES
UGVEX = MODE POSITIONS AND SLOPES OF THE NODAL POINTS
PHI = STATE TRANSITION MATRICES FOR EACH MODE
GAMMA = INPUT TRANSITION MATRIX
A = DIAGONAL MATRIX CONSISTING OF PHI

```

C B = INPUT MATRIX OF GAMMA AND CONTROL SLOPES	SPA00500
C DAMP = DAMPING FACTOR	SPA00510
C SAMPT = SAMPLING TIME	SPA00520
C IMPLSE = IMPULSE INPUT FUNCTION	SPA00530
C MIN = NUMBER OF MINUTES SYSTEM WILL BE OBSERVED	SPA00540
C SMODE = NUMBER OF STARTING MODE (INT)	SPA00550
C MODE = NUMBER OF MODES (INT)	SPA00560
C EMODE = NUMBER OF THE LAST MODE (INT)	SPA00570
C NODE = NUMBER OF THE NOISE INPUT MODE (INT)	SPA00580
C *** NOISE SLOPE LOCATIONS IN DATA MATRIX ***	SPA00590
C ROWN1 = X-SLOPE LOCATION	SPA00600
C ROWN2 = Y-SLOPE LOCATION	SPA00610
C ROWN3 = Z-SLOPE LOCATION	SPA00620
C C = OUTPUT MATRIX FOR Y	SPA00630
C IDENT = IDENTITY MATRIX	SPA00640
C RMN = MEASUREMENT NOISE COVARIANCE MATRIX	SPA00650
C QPN = PLANT NOISE COVARIANCE MATRIX	SPA00660
C PNVARX = PLANT NOISE X-SLOPE VARIANCE	SPA00670
C PNVARY = PLANT NOISE Y-SLOPE VARIANCE	SPA00680
C PNVARZ = PLANT NOISE Z-SLOPE VARIANCE	SPA00690
C MNVARX = MEASUREMENT NOISE X-SLOPE VARIANCE	SPA00700
C MNVARY = MEASUREMENT NOISE Y-SLOPE VARIANCE	SPA00710
C MNVARZ = MEASUREMENT NOISE Z-SLOPE VARIANCE	SPA00720
C ISEED = INITIALIZATION FOR RANDOM NUMBER GENERATOR	SPA00730
C RNDM = RANDOM NUMBERS USED FOR WHITE NOISE IN MEASUREMENTS AND IN PLANT FORCES	SPA00740
C BN = B MATRIX TO MULTIPLY NOISE DISTURBANCES	SPA00750
C	SPA00760
C	SPA00770
C	SPA00780
C	SPA00790
C ***** SAMPLE OF SPACE EXEC FILE *****	SPA00800
C	SPA00810
C THIS FILE MUST BEGIN IN COLUMN 1 AND RUN WITH THE FOLLOWING SEQUENCE FOR THE INITIAL RUN OF THE PROGRAM:	SPA00820
C	SPA00830
C	SPA00840
C FORTVS SPACE (COMPILES PROGRAM)	SPA00850
C SPACE (EXECUTES EXEC FILE)	SPA00860
C LOAD SPACE (START (LOADS AND EXECUTES PROGRAM)	SPA00870
C	SPA00880
C SUBSEQUENT PROGRAM RUNS CAN ELIMINATE "FORTVS SPACE" IF NO CHANGES HAVE BEEN MADE TO THE PROGRAM, AND CAN ELIMINATE RUNNING THE EXEC FILE.	SPA00890
C	SPA00900
C	SPA00910
C	SPA00920
C FI 4 DISK THESIS INPUT (PERM)	SPA00930
C FI 8 DISK UTILITY DATA (RECFM VS BLOCK 133 PERM)	SPA00940
C FI 11 DISK CNTRL OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM)	SPA00950
C FI 13 DISK GAMMA OUTPUT (RECFM VS BLOCK 133 PERM)	SPA00960
C FI 14 DISK MODE OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM)	SPA00970
C FI 16 DISK COST OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM)	SPA00980
C FI 17 DISK PRT OUTPUT (RECFM F BLOCK 80 LRECL 80 PERM)	SPA00990
C FI 18 DISK ERROR DATA (RECFM F BLOCK 80 LRECL 80 PERM)	SPA01000
C FI 19 DISK END FILE (RECFM F BLOCK 80 LRECL 80 PERM)	SPA01010
C FI 20 DISK GMAT FILE (RECFM F BLOCK 80 LRECL 80 PERM)	SPA01020
C	SPA01030
C	SPA01040
C	SPA01050

```

PARAMETER (JR=5243, KR=5397, MR=262139) SPA01060
C SPA01070
C SPA01080
C MIN = 15.0 SPA01090
C SPA01100
C WT=1.0D00 SPA01110
C PI = 4.0D0 * ATAN(1.0D0) SPA01120
C ***** SPA01130
C READ LAMA AND UGVEX MATRICIES ***** SPA01140
C ***** SPA01150
C ***** SPA01160
C ***** SPA01170
C CALL EXCMS ('CLRSCRN') SPA01180
C WRITE(6,1008) SPA01190
C WRITE(6,*) ' READING LAMA AND UGVEX MATRICIES' SPA01200
C WRITE(6,*) ' SPA01210
C THIS SECTION READS THE LAMA VECTOR AND THE UGVEX SPA01220
C MATRIX AND STORES THEM IN MEMORY FOR FURTHER RECALL OF SPA01230
C DESIRED LOCATION DATA. SPA01240
C SPA01250
C READ(4,1001) NAM SPA01260
C READ(4,1002)(LAMA(I),I=1,100) SPA01270
C READ(4,1001) NAM SPA01280
C DO 5 J = 1,100 SPA01290
C READ(4,1002)(UGVEX(I,J),I=1,684) SPA01300
5 CONTINUE SPA01310
C SPA01320
1001 FORMAT(1X,A6) SPA01330
1002 FORMAT(1X,8E15.8) SPA01340
1008 FORMAT(1X,///) SPA01350
C SPA01360
500 CALL EXCMS ('CLRSCRN') SPA01370
C SPA01380
C ***** STARTING MODE NUMBER ***** SPA01390
C ** SMODE 7 TO 100 (INTEGER) ***** SPA01400
C SMODE= 17 SPA01410
C SPA01420
C WRITE (16,700) SMODE SPA01430
700 FORMAT (' ','STARTING MODE NUMBER: ',I2) SPA01440
C SPA01450
C ***** NUMBER OF MODES TO SCAN ***** SPA01460
C ** MODE 1 TO 93 (INTEGER) SPA01470
C SPA01480
C MODE=3 SPA01490
C SPA01500
C EMODE = SMODE + MODE - 1 SPA01510
C SPA01520
C WRITE (16,701) MODE SPA01530
701 FORMAT (' ','NUMBER OF MODES SCANNED: ',I2) SPA01540
C SPA01550
C ***** NOISE INPUT POSITION ***** SPA01560
C ** NODE 1 TO 114 (INTEGER) (IF 0 THEN NO NOISE INPUT) SPA01570
C NODE= 8 SPA01580
C SPA01590
C WRITE (16,702) NODE SPA01600
702 FORMAT (' ','NOISE NODE LOCATION: ',I5) SPA01610

```

```

C ***** START AND STOP FOR PLANT SPA01620
C SN=17 SPA01630
C FN=4 SPA01640
C NS=SN*2-1 SPA01650
C NF=SN*2+2*FN-2 SPA01660
C WRITE (16,899) SN,FN SPA01670
899 FORMAT (' ','PLANT -- SN= ',I5,' FN= ',I5) SPA01680
C ***** SAMPLING TIME ***** SPA01690
C ** SAMPT MUST BE LESS THAN OR EQUAL TO SAMPTM ** SPA01700
C SAMPT = 0.05 SPA01710
C SAMPTM = ((2.0D0*PI)/SQRT(LAMA(EMODE)))/1.0D01 SPA01720
C IF (SAMPT.GE.SAMPTM) THEN SPA01730
C SAMPT=SAMPTM SPA01740
C ENDIF SPA01750
C SPA01760
C WRITE (16,900) MIN SPA01770
900 FORMAT (' ',2X,'MIN: ',F8.3) SPA01780
C SPA01790
C WRITE (16,703) SAMPT, SAMPTM SPA01800
703 FORMAT (' ','SAMPLING TIME: ',D12.4,2X,'SAMPTM= ',D15.8) SPA01810
C SPA01820
C ***** DAMPING FACTOR ***** SPA01830
C ** DAMP 0.0 TO 1.0 (REAL*8) SPA01840
C DAMP=.01 SPA01850
C SPA01860
C WRITE (16,704) DAMP SPA01870
704 FORMAT (' ','DAMPING FACTOR: ',D12.4) SPA01880
C SPA01890
C NO=3 SPA01900
C *** PLANT NOISE VARIANCE *** SPA01910
C ** PNVARX, PNVARY, PNVARZ GT 0.0 SPA01920
C SPA01930
C SPA01940
C SF=1.0D0 SPA01950
C SF1=2.5D06 SPA01960
C PNVARX=1.0D00*SF1 SPA01970
C PNVARY=1.0D00*SF1 SPA01980
C PNVARZ=1.0D00*SF1 SPA01990
C SPA02000
C SPA02010
C SPA02020
C *** MEASUREMENT NOISE VARIANCE *** SPA02030
C ** MNVARX, MNVARY, MNVARZ GT 0.0 SPA02040
C MNVARX=1.0D-03 *SF SPA02050
C MNVARY=1.0D-03 *SF SPA02060
C MNVARZ=1.0D-03 *SF SPA02070
C SPA02080
C SPA02090
C WRITE (16,711) SPA02100
711 FORMAT(' ','PLANT NOISE VARIANCE: ') SPA02110
C WRITE (16,712) SPA02120
712 FORMAT(' ',6X,'PNVARX',13X,'PNVARY',13X,'PNVARZ') SPA02130
C WRITE (16,713) PNVARX, PNVARY, PNVARZ SPA02140
713 FORMAT(' ',2X,E15.8,2X,E15.8,2X,E15.8) SPA02150
C WRITE(16,714) SPA02160
714 FORMAT(' ','MEASUREMENT NOISE: ') SPA02170

```

```

    WRITE(16,715) SPA02180
715  FORMAT(' ',6X,'MNVARX',13X,'MNVARY',13X,'MNVARZ') SPA02190
      WRITE(16,713) MNVARX,MNVARY,MNVARZ SPA02200
C
510  CALL EXCMS ('CLRSCRN') SPA02210
      WRITE (6,1008) SPA02220
      WRITE (6,*) ' PROGRAM RUNNING' SPA02230
C
C  ***** NOISE INPUT LOCATION ***** SPA02240
C
C  ***** SPA02250
C  ***** SPA02260
C  ***** SPA02270
C  ***** SPA02280
C  ***** SPA02290
C  ***** SPA02300
C  ***** SPA02310
C
C  ***** SPA02320
C  ***** SPA02330
C  ***** SPA02340
C
C  DO 54 K = 1, 200 SPA02350
      X(K) = 0.0 SPA02360
54    CONTINUE SPA02370
C
C  DO 60 I=1,3 SPA02380
      DO 58 J=1,3 SPA02390
      RMN(I,J)=0.0 SPA02400
      QPN(I,J)=0.0 SPA02410
58    CONTINUE SPA02420
60    CONTINUE SPA02430
C
C  RMN(1,1)=MNVARX**2.0 SPA02440
  RMN(2,2)=MNVARY**2.0 SPA02450
  RMN(3,3)=MNVARZ**2.0 SPA02460
  QPN(1,1)=PNVARX**2.0 SPA02470
  QPN(2,2)=PNVARY**2.0 SPA02480
  QPN(3,3)=PNVARZ**2.0 SPA02490
C
C  ***** SPA02500
C  ***** SPA02510
C
C  ***** SPA02520
C  ***** SPA02530
C  ***** SPA02540
C  ***** SPA02550
C  ***** SPA02560
C
C  CALL STMTRX(EMODE,SMODE,SAMPT,DAMP,PHI,GAMMA,A,B,LAMA,UGVEX,C, SPA02570
+    ROWN1,ROWN2,ROWN3,BN) SPA02580
C
C
C  WRITE (6,1008) SPA02590
  WRITE(6,*) ' EXIT STMTRX - - - PRE-LOOP KALMAN' SPA02600
C
C
C  WRITE (6,*) ' COMPUTING C TIMES SF FOR NEW C' SPA02610
C
C  WRITE (16,1008) SPA02620
  DO 11000 I=1,200 SPA02630
    DO 10900 J=1,NO SPA02640
      C(J,I)= C( J,I)*SF SPA02650
10900  CONTINUE SPA02660
11000 CONTINUE SPA02670
C

```

```

C     *** PRE-LOOP PORTION OF KALMAN FILTER           SPA02740
C                                                 SPA02750
C                                                 SPA02760
C                                                 SPA02770
C     JK=SMODE*2-2                                     SPA02780
C     M2=2*MODE                                       SPA02790
C                                                 SPA02800
C                                                 SPA02810
C                                                 SPA02820
C                                                 SPA02830
C     *****                                           SPA02840
C     M2=2*MODE                                     SPA02850
C     JP=2*SMODE-1                                  SPA02860
C     JQ=2*EMODE                                     SPA02870
C                                                 SPA02880
C     DO 8813 I=1,3                                 SPA02890
C     EX(I)=0.0                                     SPA02900
8813  CONTINUE                                     SPA02910
C                                                 SPA02920
C                                                 SPA02930
C                                                 SPA02940
C                                                 SPA02950
C     *****      THIS SECTION COMPUTES THE STATE UPDATE  SPA02960
C     *****                                           SPA02970
C     ESS =0.0                                       SPA02980
C     COUNT = 0                                     SPA02990
C     ENERGY = 0.0DO                                SPA03000
C     TIME = 0.0                                     SPA03010
C     CGN=0.0                                       SPA03020
C     CTG=0.0                                       SPA03030
C     *****      SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE  SPA03040
C     *****      THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED  SPA03050
C     WRITE (6,1008)                                SPA03060
C     WRITE (6,*)          START STATE UPDATE          SPA03070
C     LOOP = INT((MIN*60.0)/SAMPT)                  SPA03080
C     PRT= (DBLE(LOOP))/30.0                         SPA03090
C     PRT1=(DBLE(LOOP))/50.00                        SPA03100
C     CTT=0.0                                       SPA03110
C                                                 SPA03120
C     DO 400 L = 0, LOOP                           SPA03130
C     TIME = DBLE(L)*SAMPT                         SPA03140
C                                                 SPA03150
C     IF(L.EQ.0)THEN                                SPA03160
C       IMPLSE =(1.0D06*SF1)/(DSQRT(SAMPT))        SPA03170
C     ELSE                                           SPA03180
C       IMPLSE = 0.0DO                             SPA03190
C     ENDIF                                         SPA03200
C                                                 SPA03210
C     TO=0.0                                         SPA03220
C     *****      RANDOM NUMBER GENERATOR           SPA03230
C                                                 SPA03240
C     DO 101 I=1,6                                 SPA03250
C     ISEED=MOD(ISEED*JR+KR,MR)                    SPA03260
C     RND1=(DBLE(ISEED)+0.5D00)/DBLE(MR)          SPA03270
C     ISEED=MOD(ISEED*JR+KR,MR)                    SPA03280
C     RND2=(DBLE(ISEED)+0.5D00)/DBLE(MR)          SPA03290

```

```

101      RNDM(I)=DSQRT(-2.0*DLOG(RND1))*DCOS(6.2831853D00*RND2)      SPA03300
C      CONTINUE
C      *****
C      CTT=CTT+1.0
C      **** START OF STATE UPDATE ***
C      ***
C      COMPUTE AX0200 = A0200 X 200 * X0200
C      ***
C      ***
C      COMPUTE AXH = A*XH
C      JK=SMODE*2-2
C      JP=JK+1
C      JQ=2*EMODE
C      DO 5015 I=NS,NF
C          SUM=0.0
C          DO 5010 K=NS,NF
C              SUM=SUM+A(I,K)*X(K)
5010      CONTINUE
C          AX(I)=SUM
5015      CONTINUE
C      ***
C      COMPUTE WD03
C      WD(1)=PNVARX*RNDM(1)*TO+IMPLSE
C      WD(2)=PNVARY*RNDM(2)*TO
C      WD(3)=PNVARZ*RNDM(3)*TO
C      ***
C      COMPUTE BNWD0200 =BN0200 X 3 * WD03
C      DO 5035 I=NS,NF
C          SUM=0.0
C          DO 5030 K=1,3
C              SUM=SUM+BN(I,K)*WD(K)
5030      CONTINUE
C          BNWD(I)=SUM
5035      CONTINUE
C      ***
C      COMPUTE X0200 =AX0200 + BNWD0200
C      DO 5040 I=NS,NF
C          X(I)= AX(I) + BNWD(I)
C          IF (DABS(X(I)).LT. 1.0D-60) THEN
C              X(I)=1.0D-60
C          END IF
C      5040      CONTINUE
C      ****
C      **** START OF KALMAN FILTER ***
C

```

```

JK=SMODE*2-2 SPA03860
JP=JK+1 SPA03870
JQ=2*EMODE SPA03880
M2=2*MODE SPA03890
SPA03900
SPA03910
SPA03920
SPA03930
SPA03940
SPA03950
SPA03960
SPA03970
SPA03980
SPA03990
SPA04000
SPA04010
SPA04020
SPA04030
SPA04040
SPA04050
SPA04060
SPA04070
SPA04080
SPA04090
SPA04100
SPA04110
SPA04120
SPA04130
SPA04140
SPA04150
SPA04160
SPA04170
SPA04180
SPA04190
SPA04200
SPA04210
SPA04220
SPA04230
SPA04240
SPA04250
SPA04260
SPA04270
SPA04280
SPA04290
SPA04300
SPA04310
SPA04320
SPA04330
SPA04340
SPA04350
SPA04360
SPA04370
SPA04380
SPA04390
SPA04400
SPA04410

C
JL=JQ+1
DO 8888 I=1,NO
SUM=0.0
DO 8887 K=JL,NF
SUM=SUM+C(I,K)*X(K)
CONTINUE
EX(I)=SUM*SUM*SAMPT+EX(I)
CONTINUE
C
CGN=CGN+1.0
IF (CTT.EQ.1.0.OR.CGN.GT.PRT) THEN
C
WRITE (16,1008)
WRITE (16,*) 'TIME = ', TIME
C
DO 380 I=JP , JQ
WRITE (16,4500) I,X(I)
CONTINUE
4500 FORMAT (' ',2X,'X(' ,I4,')= ',D15.8)
C
CGN=0.0
END IF
C
C
400 CONTINUE
WRITE (11,*) 'SMODE = ', SMODE
WRITE (11,*) 'EMODE = ', EMODE
WRITE (11,*) 'SN = ', SN
WRITE (11,*) 'FN = ', FN
C
JL=JQ+1
DO 9499 I=1,NO
WRITE (11,*) 'EX ',I,' ', EX(I)
CONTINUE
C
C
C
C
CALL EXCMS ('CLRSCRN')
WRITE (6,1008)
C
599 STOP
END
C
C
C
C
C

```

```

***** THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH
***** OF THE 100 MODES
***** SUBROUTINE STMTRX(EMODE,SMODE,T,D,PHI,GAMMA,A,B,LAMA,UGVEX,C,
+ ROWN1,ROWN2,ROWN3,BN)
REAL*8 WN,GMA,PHI(2,2,100),GAMMA(2,100),EGT,T,COSW1T,SINW1T
REAL*8 W1,D,A(200,200),B(200,3),C(9,200),BN(200,3)
REAL LAMA(100),UGVEX(684,100)
INTEGER SMODE,R,EMODE,JJ,KK,ROWN1,ROWN2,ROWN3, NN(9), N9, NO
WRITE (6,*) 'INSIDE STMTRX -- COMPUTE WN, GMA, EFT, W1'
DO 600 I = 1 ,100
  WN = DBLE(SQRT(LAMA(I)))
  GMA = D*WN/2.0
  EGT = DEXP( -GMA*T)
  W1 = DSQRT((WN**2)-(GMA**2))
  COSW1T = DCOS(W1*T)
  SINW1T = DSIN(W1*T)
IF(WN.EQ.0)THEN
  PHI(1,1,I) = EGT*COSW1T
  PHI(1,2,I) = T
  PHI(2,1,I) = 0
  PHI(2,2,I) = EGT*COSW1T
  GAMMA(1,I) = 0
  GAMMA(2,I) = 0
ELSE
  PHI(1,1,I) = EGT*(COSW1T + (GMA*(W1**(-1)))*SINW1T)
  PHI(1,2,I) = (W1**(-1))*EGT*SINW1T
  PHI(2,1,I) = -(WN**2)*(W1**(-1))*EGT*SINW1T
  PHI(2,2,I) = EGT*(COSW1T - (GMA*(W1**(-1)))*SINW1T)
  GAMMA(1,I)=(WN**(-2))*((1.0D0-EGT*COSW1T -EGT*(GMA/W1)*SINW1T)

```

```

C           GAMMA(2,I) = (W1**(-1))*EGT*SINW1T           SPA04970
C           SPA04980
C           SPA04990
C           SPA05000
C           SPA05010
C           SPA05020
C           SPA05030
C           SPA05040
C           SPA05050
C           SPA05060
C           SPA05070
C           SPA05080
C           SPA05090
C           SPA05100
C           SPA05110
C           SPA05120
C           SPA05130
C           SPA05140
C           SPA05150
C           SPA05160
C           SPA05170
C           SPA05180
C           SPA05190
C           SPA05200
C           SPA05210
C           SPA05220
C           SPA05230
C           SPA05240
C           SPA05250
C           SPA05260
C           SPA05270
C           SPA05280
C           SPA05290
C           SPA05300
C           SPA05310
C           SPA05320
C           SPA05330
C           SPA05340
C           SPA05350
C           SPA05360
C           SPA05370
C           SPA05380
C           SPA05390
C           SPA05400
C           SPA05410
C           SPA05420
C           SPA05430
C           SPA05440
C           SPA05450
C           SPA05460
C           SPA05470
C           SPA05480
C           SPA05490
C           SPA05500
C           SPA05510
C           SPA05520

600      CONTINUE

C           WRITE (6,*) 'PHI AND GAMMA COMPUTED'
C           WRITE (6,*) ' COMPUTING A, B, BN'

C           R = 1

C           DO 610 K = 1      ,100

C           A(R,R) = PHI(1,1,K)
C           A(R,R+1) = PHI(1,2,K)
C           A(R+1,R) = PHI(2,1,K)
C           A(R+1,R+1) = PHI(2,2,K)

C           *** B MATRIX FOR MULTIPLYING CONTROL TORQUES

C           B(R,1) = GAMMA(1,K)*DBLE(UGVEX(412,K))
C           B(R,2) = GAMMA(1,K)*DBLE(UGVEX(413,K))
C           B(R,3) = GAMMA(1,K)*DBLE(UGVEX(414,K))
C           B(R+1,1) = GAMMA(2,K)*DBLE(UGVEX(412,K))
C           B(R+1,2) = GAMMA(2,K)*DBLE(UGVEX(413,K))
C           B(R+1,3) = GAMMA(2,K)*DBLE(UGVEX(414,K))

C           *** BN MATRIX FOR MULTIPLYING THE NOISE DISTURBANCES

C           BN(R,1)=GAMMA(1,K)*DBLE(UGVEX(ROWN1,K))
C           BN(R,2)=GAMMA(1,K)*DBLE(UGVEX(ROWN2,K))
C           BN(R,3)=GAMMA(1,K)*DBLE(UGVEX(ROWN3,K))
C           BN(R+1,1)=GAMMA(2,K)*DBLE(UGVEX(ROWN1,K))
C           BN(R+1,2)=GAMMA(2,K)*DBLE(UGVEX(ROWN2,K))

```

```

C BN(R+1,3)=GAMMA(2,K)*DBLE(UGVEX(ROWN3,K)) SPA05530
C SPA05540
C SPA05550
C SPA05560
C SPA05570
C SPA05580
C SPA05590
C SPA05600
C SPA05610
C SPA05620
C SPA05630
C SPA05640
C SPA05650
C SPA05660
C SPA05670
C SPA05680
C SPA05690
C SPA05700
C SPA05710
C SPA05720
C SPA05730
C SPA05740
C SPA05750
C SPA05760
C SPA05770
C SPA05780
C SPA05790
C SPA05800
C SPA05810
C SPA05820
C SPA05830
C SPA05840
C SPA05850
C SPA05860
C SPA05870
C SPA05880
C SPA05890
C SPA05900
C SPA05910
C SPA05920
C SPA05930
C SPA05940
C SPA05950
C SPA05960
C SPA05970
C SPA05980
C SPA05990
C SPA06000
C SPA06010

C R = R+2
610  CONTINUE
C WRITE (6,*) 'A, B, BN COMPUTED'
C WRITE (6,*) 'COMPUTING C'
C
C **** C MATRIX PRODUCTION ****
C NO=3
C NN(1)=418
C NN(2)=419
C NN(3)=420
C
C JJ=-1
C DO 640 I=1,100
C JJ=JJ+1
C
C DO 9127 K=1,NO
C
C KK=I+JJ
C
C N9=NN(K)
C
C C(K,KK) = DBLE(UGVEX(N9,I))
C
C KK=KK+1
C
C C(K,KK)=0.0
9127  CONTINUE
640  CONTINUE
C
C
C RETURN
C END

```

## LIST OF REFERENCES

1. Rodriguez, G., *Control Technology Development*, Large Space Systems Technology - 1980, NASA Conference Publication 2168, Vol. I, pp. 49-63, November 18-20, 1980.
2. Thomson, William T., *Theory of Vibration with Applications*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.
3. Preston, William J. , *Effects of Reduced Order Modeling on the Control of a Large Space Structure*, Masters Thesis, Naval Postgraduate School, CA, September 1988.
4. Sabby, Samuel M., *Standard Mathematical Tables*, The Chemical Rubber Co., Cleveland, OH, 1972.
5. Mendel, Jerry M., *Discrete Techniques of Parameter Estimation*, Marcel Dekker, Inc., NY, 1973.
6. Nishimura, T., "On the a priori Information in Sequential Estimation Problems," *Kalman Filtering: Theory and Application*, p. 138-145, Edited by: Sorenson, H. W., IEEE Press, Inc., New York, 1985.
7. Franklin, Gene F. and Powell, David J., *Digital Control of Dynamic Systems*, Addison-Wesley Company, Reading, MA, 1980.

## INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3. Director Naval Research Laboratory Washington, D.C. 20375	1
4. Commander Naval Space Command Attn: Code N3 Dahlgren, VA 22448-5170	1
5. Chairman, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943	1
6. Prof. Jeff B. Burl, Code 62BL Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943	1
7. Prof. H. A. Titus, Code 62TS Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943	1
8. Major W. J. Preston 7 Cary Place Fredricksburg, VA 22405	1
9. Lcdr. Bruce M. Jackson c/o Col. Jose A. Aybar (USA-Ret) 22577 Veronica Drive Salinas, CA 93943	2